Universitat de Lleida

# TREBALL FINAL DE GRAU


Estudiant: Tatiana Vilalta Vidal

Titulació: Grau en Enginyeria Informàtica

Títol de Treball Final de Grau: **Desarrollo de Partyfy, una app híbrida en React Native**

Director/a: **Antoni Granollers**

Presentació

Mes: Setembre

Any: 2019

# Índice de contenidos

# 1. Introducción

## 1.1 Motivación del trabajo

La aplicación que se desarrolla en este trabajo surge a raíz de la necesidad de un promotor de discotecas y eventos de crear una plataforma mediante la cual los usuarios podrán tener al alcance la información de ocio de las ciudades que desee y dónde las empresas podrán mostrar sus eventos (cartel, descripción y poner a la venta las entradas).

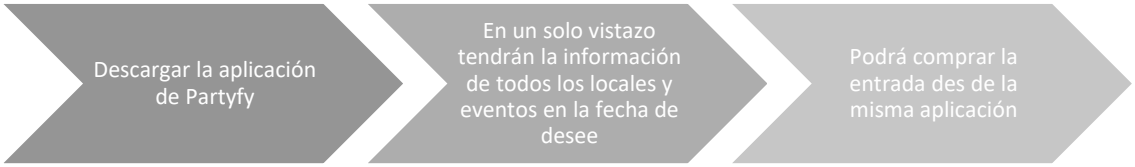
La evolución de la tecnología y sobre todo el auge de las diferentes redes sociales (Facebook, Instagram, Snapchat, Pinterest…) hacen que los usuarios tengan una cantidad de información muy elevada, pero a la vez muy dispersa.

Todos los usuarios comparten contenidos en dichas redes, y lo mismo utilizan las empresas para promocionar sus productos, eventos, ofertas etc… así conseguimos que internet tenga mucha información, pero no sabemos qué información es fiable.

La aplicación PARTYFY, ideada, diseñada y desarrollada durante el periodo de este proyecto, tiene la meta de recoger toda la información sobre eventos de ocio (conciertos, cartel de las discotecas o pubs, fiestas etc..) que generan los locales y promotores, y agruparla en una misma aplicación multiplataforma para que, si un usuario no sabe qué hacer para divertirse en una ciudad concreta, pueda escoger entre todas las posibilidades que le ofrecen los locales de ocio de dicha ciudad.

De esta forma reducimos los pasos que tiene que hacer el usuario (para cada local de la ciudad):


A los siguientes:



## 1.2 Objetivos y metodologia del trabajo

El objetivo de este trabajo consiste en crear una aplicación móvil que muestre toda la información extraída de un *backend*[1] externo. Es una información referente a eventos de una ciudad o zona mas cercanos al usuario que la tiene instalada en su dispositivo.

La metodología que se ha seguido durante el proyecto ha sido la siguiente:

1. Se ha realizado una toma de requerimientos con el promotor para conocer qué información se quiere mostrar y de qué forma.
2. Se ha realizado un estudio de las opciones que se ofrecen actualmente para elaborar la aplicación (Qué es mejor, ¿una *App* nativa de Android, un *framework*[2] que pueda facilitar la adaptación a otras plataformas…?).
3. Se ha realizado un estudio de mercado para conocer qué aplicaciones existen actualmente, qué soluciones pueden dar a los requisitos del proyecto y si ofrecen alguna otra solución que también pueda funcionar en nuestra aplicación.
4. Se ha desarrollado la aplicación en el lenguaje escogido y en la plataforma que mejor se ha adaptado a las necesidades del cliente, en este caso React Native y Expo han sido las tecnologías y herramientas escogidas, en el apartado de tecnologías del proyecto las explicaré.
5. Se ha realizado una batería de pruebas para evaluar la correcta funcionalidad de la aplicación y también examinar su usabilidad, en diferentes puntos del proyecto. Estas pruebas se han realizado tanto para evaluar el prototipo inicial de la aplicación como para examinar su usabilidad una vez terminada la parte de programación.

Aprovechando los conocimientos adquiridos en las asignaturas de segundo y tercer año del grado, Interacción Persona Ordenador (IPO) y Diseño Centrado en el Usuario (DCU), y destacando la importancia del usuario en este tipo de aplicaciones, se ha elaborado todo el proyecto consultando las directrices del diseño centrado en el usuario pero sin seguirlas todas por falta de tiempo. Se han seguido los pasos correspondientes para elaborar un informe de usabilidad, hecho que incluye la elaboración de una guía de estilo para que todos los componentes sigan unas normas, un prototipo de la aplicación para diseñar todas las pantallas correspondientes, la realización de pruebas a usuarios reales en los diferentes puntos del desarrollo del proyecto que nos han permitido corregir errores de diseño y mejorar la usabilidad de las diferentes acciones… entre otras tareas.

Una vez finalizada la aplicación, el usuario tendrá a su alcance toda la información de los locales de ocio de la zona, podrá añadirla en su agenda, reservar / comprar las entradas, compartir todo este contenido en sus redes sociales o mediante las aplicaciones de mensajería, así como utilizar el navegador de su dispositivo para ir al local del evento.

[1] *Backend:* El *backend* es la parte del desarrollo web que se encarga de que toda la lógica de un sitio web o aplicación móvil funcione correctamente. Se trata de un conjunto de acciones que suceden en una aplicación, pero el usuario no vé (p.e. la comunicación con el servidor).

[2] *Framework:* Un *framework* és una estructura conceptual y tecnológica de soporte definido, normalmente contiene módulos de *software* concretos que se pueden utilizar de base para la organización y el desarrollo.

Los objetivos claves del proyecto, agrupados en puntos clave, son los siguientes:

- → Desarrollar una aplicación móvil que cumpla con los requisitos del cliente .
- → Aprender una forma de programar dicha aplicación diferente de la estudiada en el grado de Informática para ver de esta forma las diferencias entre ellas, los puntos fuertes o las desventajas.
- → Aprender los conocimientos de una nueva tecnología, para estar al día de las últimas novedades de este sector.
- → Profundizar en la forma de trabajar y procesos que requiere el Diseño y desarrollo centrado en el usuario.
- → Agilizar mi manera de programar, así como afrontar un proyecto propio.
- → Aprender a organizar las tareas y fases del proyecto, así como su temporalización. Para ello he utilizado metodologías ágiles como Scrum[3] (utilizadas en algunas de las asignaturas del grado) y la metodología MoSCoW[4] para priorizar tareas y requisitos.

## 1.3 Tareas y estructura del documento

Las tareas que se han realizado en este proyecto y que a la vez definen las diferentes partes de este documento son las siguientes:

- → **Toma de requisitos.**
  Reuniones con el promotor previas al proyecto para definir los requisitos de éste y las necesidades específicas del cliente. De estos encuentros se han extraído los puntos clave necesarios para desarrollar la aplicación, así como las peticiones de las características concretas de algunas pantallas.

- → **Estudios previos.**
  - Investigación de aplicaciones existentes.
    Se ha realizado un estudio de la posible competencia, aplicaciones o sitios webs similares. También se han analizado sus características y puntos fuertes y débiles.
  - Estudio de las diferentes opciones de desarrollo.
    Se ha hecho una valoración previa sobre la opción de hacer o no una aplicación nativa. Se han valorado las diferentes opciones (Aplicación nativa, Aplicación híbrida o *framework, webapp*) y finalmente se ha escogido el desarrollo de una aplicación adaptable a los diferentes dispositivos mediante el uso de React Native.

- → **Planificación de las tareas a realizar.**

[3] *Scrum:* método para trabajar en equipo a partir de iteraciones o Sprints. Es una metodología ágil, por lo que su objetivo es controlar y planificar proyectos con un gran volumen de tareas. Se suele planificar por semanas. Al final de cada Sprint o iteración, se revisa el trabajo validado de la anterior semana, en función de esto, se priorizan y planifican las actividades del siguiente. La metodología Scrum se centra en ajustar sus resultados y responder a las exigencias reales y exactas del cliente.

[4] *Método MoSCoW:* técnica de priorización de requisitos basada en el hecho de que destacar aquellos requisitos que permiten darle un mayor valor al sistema, lo que permite enfocar los trabajos de manera más eficiente. Definen 4 características: M – Must, S – Should, C – Could, W – won't , según se la tarea 'Es obligatorio que esté', 'Debería estar', 'Podría estar' pero no es obligatorio y finalmente 'No estará' al menos de momento.

Elaboración de una lista ordenada de las diferentes tareas correspondientes al proyecto junto con su temporalización. Para evaluar estos puntos se ha usado un diagrama de Gantt[5] que ha facilitado ver la evolución de tiempo de las tareas, así como la diferencia de tiempo entre la estimación inicial y el tiempo que finalmente se ha requerido para realizar la aplicación.

→ **Elaboración de los elementos imprescindibles para el desarrollo**:

- Manual de estilo de la marca.
  Dónde se especifican los colores y tipografías corporativas y otras aplicaciones.
- Elaboración de un *wireframe*[6].
  Esquema dónde se han definido de forma esquemática y muy genérica las diferentes partes de la aplicación (las pantallas).
- Elaboración de un prototipo.
  Mediante herramientas de diseño (Photoshop e Illustrator) se ha realizado una representación gráfica que se usará de base para la programación de la aplicación. También se ha incorporado una herramienta web para simular la interacción de las pantallas.
- Pruebas de usabilidad del diseño de la aplicación.
  Se ha usado el prototipo anterior para explicar a los usuarios escogidos el funcionamiento de la aplicación y éstos podrán evaluar su diseño y la interacción.
  Una vez ejecutadas las pruebas, se han realizado los cambios necesarios basados en las respuestas de los usuarios para seguir las pautas del diseño centrado en el usuario.

→ **Aprendizaje del framework.**
Se han realizado dos cursos dentro de la plataforma on-line de *Udemy* para aprender el uso del framework tanto en aplicaciones de escritorio como en aplicaciones móviles. Los cursos son los siguientes: React JS + Redux + ES6. Completo ¡De 0 a experto! y Desarrollo móvil con React Native, Expo y Firebase.
En ellos se ha aprendido las nociones más importantes para poder realizar la aplicación correspondiente.

→ **Programación de la aplicación.**
Parte más extensa del trabajo junto con la realización de los cursos. Es la parte básica del proyecto puesto que corresponde al desarrollo como tal de la aplicación.

→ **Pruebas.**
Durante el desarrollo de la aplicación y sobre todo en el proceso final se han realizado dos tipos de pruebas:

- Test de funcionalidad.
  Pruebas y testeos correspondientes a la comprobación del funcionamiento de la aplicación. Estas pruebas se realizan para comprobar que la aplicación extrae los datos correctos del *backend* y los muestra siguiendo el diseño del prototipo.

[5] *Diagrama de Gantt:* herramienta gráfica cuyo objetivo es exponer el tiempo de dedicación previsto para diferentes tareas o actividades a lo largo de un tiempo total determinado.
[6] *Wireframe:* boceto donde se representa visualmente, de una forma muy sencilla y esquemática la estructura de una página web o aplicación.

- Test de UX [7]y de usabilidad[8].
  Pruebas correspondientes a la comprobación final de los usuarios, para poder verificar que su interacción con la aplicación es la correcta y la visualización de los diferentes elementos de la aplicación son inteligibles.

De todas las pruebas realizadas se ha recogido el *feedback* correspondiente para realizar las mejoras pertinentes en la aplicación y en su lógica. En algunos momentos se ha requerido solicitar modificaciones de programación del *backend* externo para mejorar la interacción con la aplicación.

→ **Conclusiones del trabajo y agradecimientos.**
Parte final del trabajo, donde incluye la redacción de esta memoria, así como una reflexión sobre el trabajo realizado. También se explican los diferentes errores cometidos y las tareas pendientes por hacer.

→ **Recopilación y redacción de la bibliografía del trabajo.**
Redacción del sumario de enlaces y fuentes consultadas para la realización y redacción de esta memoria, para la investigación inicial de los diferentes ítems del proyecto y finalmente fuentes consultadas durante el desarrollo de la aplicación móvil.

## 1.4 Temporalización del proyecto

Una vez enumeradas las tareas que se han realizado en el marco del proyecto, se evaluó su complejidad y se realizó la siguiente estimación de tiempo, medida en semanas:

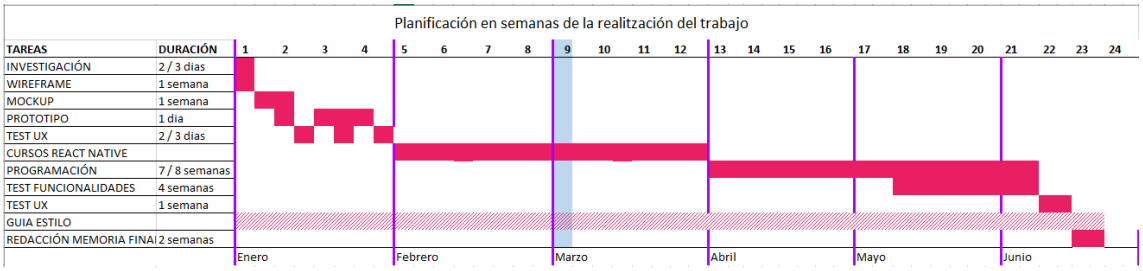



*Ilustración 1. Temporalización inicial del proyecto*

Finalmente, la temporalización del trabajo ha sido la siguiente:

[7] *User Experience (UX):* La experiencia de usuario es todo aquello que una persona percibe al interactuar con un producto o servicio. Se consigue una buena UX si la base del proyecto es el diseño de productos, servicios o aplicaciones útiles y usables. Todas las tareas que se basan en esto logran que el usuario se sienta satisfecho y feliz con el producto que esta usando (más información: http://mpiua.invid.udl.cat/user-experience/)

[8] *Usabilidad*: Coloquialmente suele definirse usabilidad como la propiedad que tiene un determinado sistema para que sea "fácil de usar o de utilizar y de aprender" tratándose de una propiedad que no es sólo aplicable a los sistemas software, sino que es aplicable a los elementos de la vida cotidiana. (más información: http://mpiua.invid.udl.cat/usabilidad/definicion/).

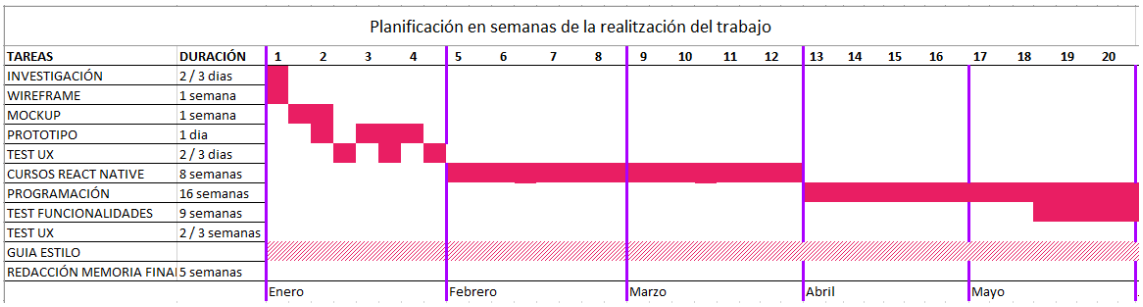



*Ilustración 2. Temporalización final del proyecto Enero-Mayo*

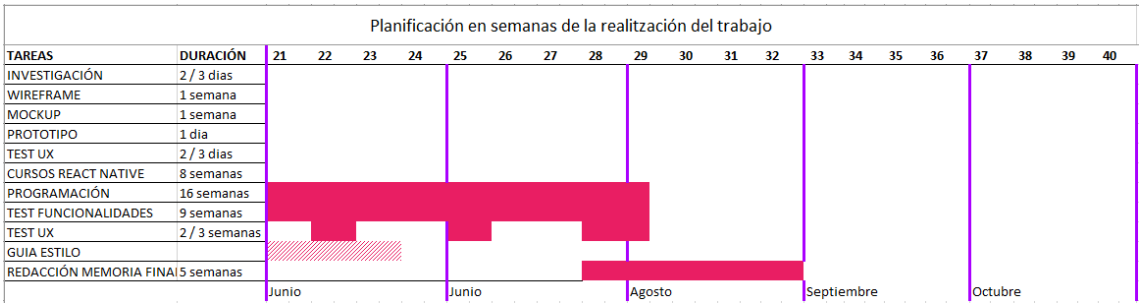



*Ilustración 3. Temporalización final del proyecto Junio-Agosto*

El proyecto ha durado mas de lo previsto, concretamente 2 meses. La propia complejidad de la aplicación ha hecho que el tiempo estimado de desarrollo haya augmentado retrasando así la planificación inicial.

Otro punto clave causante del retraso es el haber añadido una pasarela de pago. Inicialmente el objetivo era trabajar con la plataforma de *Verse*[9] *for business* pero como durante el desarrollo de la aplicación aún no había salido a la luz dicha plataforma, se cambió a usar la pasarela de pago de *Stripe*[10].

También se han incorporado más pruebas de usabilidad y diseño en la parte final del desarrollo. El motivo de este aumento ha sido testear las nuevas pantallas, así como añadir *feedback* de posibles usuarios para mejorar la interacción con la aplicación. Estas pruebas han conseguido que se mejorase en algunos puntos el diseño y de esta forma retroalimentar la cadena de desarrollo con sugerencias de los usuarios.

Este cambio, otras modificaciones de diseño y la dificultad de combinar el trabajo con el entorno laboral han hecho que la aplicación se retrase más de lo esperado.

[9] *Verse:* Aplicación para pagar y recibir pagos des del móvil.

[10] *Stripe:* es una compañía electrónica que tiene un software para que individuos y negocios realicen pagos por internet. Proporcionan una API para poder implementar su pasarela de pago en cualquier aplicación.

## 1.5 Presupuesto

El precio de mercado del desarrollo de esta aplicación podría ser el siguiente:

| ITEM | HORAS INVERTIDAS | PRECIO/HORA | PRECIO TOTAL |
|---|---|---|---|
| **Diseño de la aplicación** | 40h | 30€ | 1.200€ |
| **Desarrollo** | 120h | 30€ | 3.600€ |
| **Testing** | 24h | 30€ | 720€ |
| | | **TOTAL** | **5.520€** |

Este presupuesto sería un ejemplo si esta aplicación la presupuestara una empresa de desarrollo de aplicaciones, se entiende que el conocimiento del *framework* se da por hecho, y el desarrollo del *backend* de dicha aplicación no está incluido en este presupuesto ya que no es una tarea contemplada en el trabajo.

Los gastos secundarios como puede ser la luz, internet etc… están calculados en el precio/hora del presupuesto.

El objetivo de esta aplicación no es ser comercializada a cada local de ocio, si no que todos los locales puedan publicitar en ella sus eventos y entradas. Por ese motivo la idea de comercialización del proyecto seria una tarifa plana mensual o anual para cada uno de los locales que quieran formar parte de ella. Un ejemplo de subscripción podría ser de 15€/mes.

# 2. Estudios previos

## 2.1 Modelos de desarrollo (Lenguajes y plataformas)

Una vez detallados los requisitos del promotor, que estarán explicados en el punto 3.1, la cuestión más importante que surgió fue escoger cómo desarrollar la aplicación en lo que a tecnologías se refiere.

Durante el grado hemos trabajado con Android puro para realizar plataformas móviles y también hemos profundizado en sus prestaciones y sensores. Pero, en el momento de afrontar el desarrollo había un punto importante: el dispositivo en el que se va a visualizar.

Si analizamos las gráficas siguientes:

*"2 DE CADA 3 MINUTOS QUE NAVEGAMOS POR INTERNET LO HACEMOS DESDE DISPOSITIVOS MÓVILES"*

*Fuente:* iabspain


*Ilustración 4. Cuota de mercado según el SO a nivel mundial - Julio 2019*

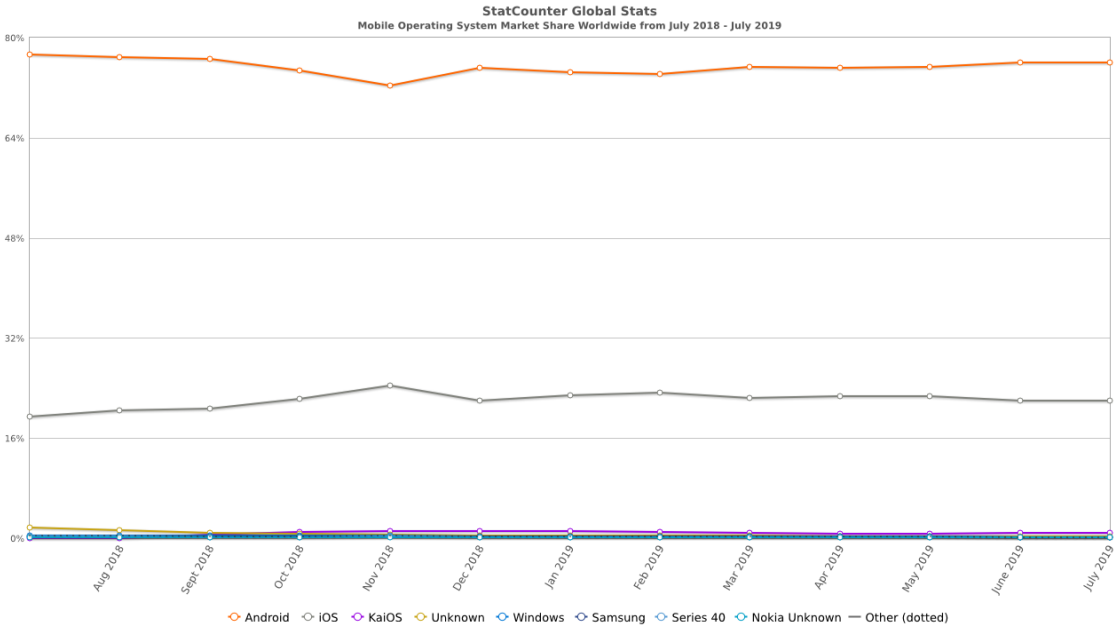



*Ilustración 5. Cuota de mercado según el SO entre junio - julio 2019*

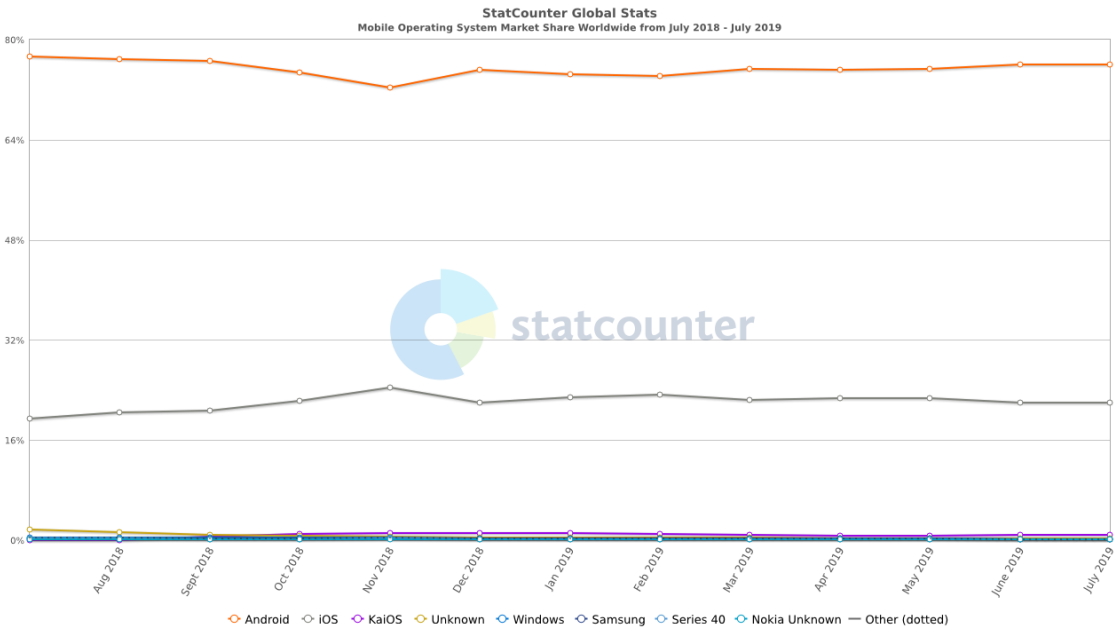



*Ilustración 6. Cuadro de mercado según el SO a nivel mundial entre junio 2018 - julio 2019*


*Ilustración 7. Cuota de mercado en España durante el mes de Julio 2019*

*Fuente:* Global Stats

La información que obtenemos de los datos anteriores nos confirma que los usuarios cada vez usamos más los dispositivos móviles para realizar todo tipo de tareas y sobre todo para conectarnos a Internet y consultar información.

También podemos ver que, por mucho que el porcentaje de dispositivos móviles con sistema operativo Android sea muy elevado (dato lógico puesto que es el que utilizan la mayor parte de fabricantes de dispositivos como Samsung, Motorola, Huawei…), iOS también está en auge y cada vez más usuarios se deciden por los dispositivos de la marca estadounidense. Estos datos han sido muy importantes en el momento de decidir la metodología a utilizar en el desarrollo de la aplicación móvil.

Una vez hecho este análisis se plantea una duda ¿Desarrollar la aplicación de forma nativa, por ejemplo, para Android, y así dejar un porcentaje de mercado sin ser un público objetivo? O ¿Desarrollar una aplicación que fuese compatible con todos los dispositivos ahorrando así en tiempo de desarrollo, aunque eso suponga un aprendizaje extra diferente de lo aprendido durante el grado?

Antes de tomar esa decisión, se ha hecho un estudio de las diferentes posibilidades:

- → Aplicación Nativa
- → Aplicación Web
- → Aplicación Híbrida

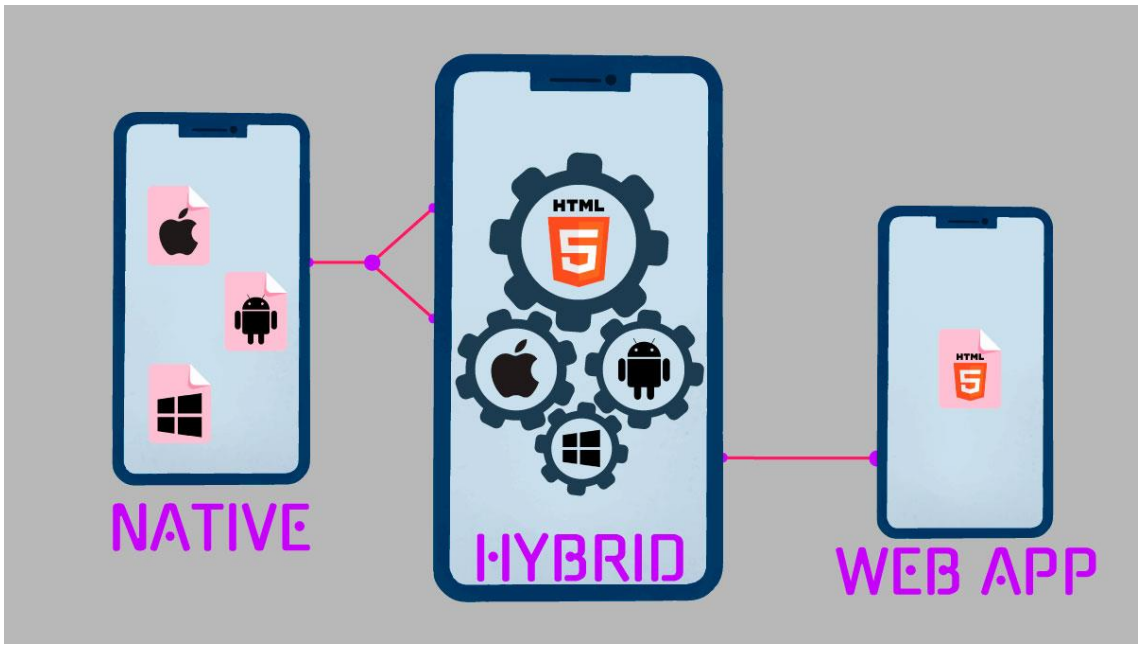



*Ilustración 8. Comparación entre Aplicaciones*

Se ha analizado cada posibilidad con sus ventajas y desventajas, así como se ha analizado qué tipo de desarrollo han escogido algunas de las aplicaciones y plataformas más usadas del mercado.

Según cada tipo de aplicación intervienen lenguajes diferentes y conceptos de desarrollo y diseño distintos para cada una, este es un echo que también se ha tenido en cuenta.

### 2.1.1 Aplicaciones Nativas

Las aplicaciones nativas o *client-side apps* son aquellas aplicaciones que se desarrollan específicamente para cada sistema operativo dónde se van a ejecutar. Este echo implica que, para cada uno de estos sistemas será necesario adaptar el código al lenguaje concreto con el que trabaja:

| SISTEMA OPERATIVO | LENGUAJE | PROPIETARIO |
|---|---|---|
| **iOS** | Objective-C, Swift | Apple |
| **Android** | Java, Kotlin | Google |
| **Windows Phone** | .Net, C#, Visual Basic | Microsoft |
| **Blackberry** | C, C# | RIM[11] |

*Tabla 1. Lenguajes de programación según el sistema operativo*

Son aplicaciones que, al estar pensadas para cada dispositivo en concreto, pueden acceder a sus funcionalidades y sacarles el máximo partido (como pueden ser la geolocalización GPS, el acceso a la galería de imágenes, acceso a los contactos del dispositivo o incluso a sus sensores), por ese motivo son aplicaciones con una navegación más ágil y fluida que mejora la experiencia de usuario.

Generalmente son aplicaciones que no necesitan conexión a internet (si la aplicación no lo requiere), pueden ejecutarse en segundo plano mientras el usuario está interactuando con otras aplicaciones, y lanzar *notificaciones push*[12] para notificar eventos al usuario.

Por lo general son aplicaciones más costosas, tanto en términos de tiempo como en términos económicos. El código no es reutilizable para las diferentes plataformas, lo que implica que en cada una de ellas se debe empezar la aplicación des de 0 y eso requiere un nivel de conocimientos en el desarrollo de aplicaciones elevado en los diferentes lenguajes.

Este tipo de aplicaciones se publica en las *App Store* (Google Play y Apple Store) pasando por unos exhaustivos controles y obligando al usuario a actualizar la aplicación cada vez que el desarrollador realice algún cambio. Google Play permite que las aplicaciones estén disponibles en pocas horas des de su publicación, pero el proceso de validación de la Apple Store es más lento. Este echo puede ser una ventaja hacia las aplicaciones nativas o híbridas puesto que están disponibles en una plataforma a la vista de todos los usuarios y llevan un control de calidad, aun así el tiempo de validación puede suponer un *hándicap* para el desarrollador si tiene que ser validada cada vez que se realiza una actualización.

[11] *RIM:* compañía canadiense de dispositivos inalámbricos más conocido como el fabricante y promotor del dispositivo de comunicación de mano BlackBerry.

[12] *Notificaciones push*: la tecnología push es una forma de comunicación a través de Internet en la que la petición de envío tiene origen en el servidor, es decir, son los mensajes instantáneos que el usuario recibe en el dispositivo.

Algunas de las aplicaciones que usamos comúnmente son aplicaciones nativas ya sean aplicaciones de trabajo, de redes sociales, o de ocio. A continuación, podemos ver algunos ejemplos:

|  |  |  |  |
|---|---|---|---|
| **SKYPE** | **NETFLIX** | **CANDY CRUSH** | WAZE |
| Año de creación: 2003 | Año de creación: 2010 | Año de creación: 2012 | Año de creación: 2013 |

## *2.1.2 Aplicaciones Web*

Las *Web apps* o *server-side apps* son aplicaciones que utilizan lenguajes como Javascript, HTML y CSS que se adaptan a todos los sistemas operativos y a todas las plataformas.

No es necesaria una programación especial para cada dispositivo dónde se van a ejecutar las aplicaciones puesto que éstas se ejecutan a través del navegador web de cada sistema operativo y están programadas para adaptarse a cada uno de ellos mediante el Responsive Web Design[13].

El desarrollo de estas aplicaciones es más económico que el anterior, puesto que no es necesario duplicar el trabajo para cada dispositivo, y es más versátil ya que al visualizarse en el navegador se puede utilizar en dispositivos móviles o no móviles (como por ejemplo en un ordenador).

No son aplicaciones que se descarguen a través de las App Stores, por ello no ocupan memoria en los dispositivos dónde se utilizan, ni se deben actualizar puesto que el usuario siempre visualiza la versión más reciente y actualizada.

Una gran desventaja de las web apps es que requieren que el dispositivo tenga conexión a internet para visualizar su contenido y un espacio web para guardar la información (un servidor), también suponen un gasto de datos para el usuario ya que, al conectarse a internet, toda la información correspondiente al sitio web se descarga en el dispositivo para que éste la pueda mostrar al usuario.

Como ya se ha nombrado, son aplicaciones que se visualizan en el navegador, por lo que la seguridad de estas está sujeta a dicho navegador. Este echo también limita su ejecución, a diferencia de las aplicaciones nativas, no se pueden ejecutar en segundo plano, limitando de esta forma las funciones del dispositivo a las que se puede acceder o utilizar.

Son aplicaciones que no acceden a los recursos del dispositivo, y tampoco presentan un diseño basado en las directrices de cada sistema. Por regla general son aplicaciones que, siguiendo el diseño adaptable, se desarrollan para la visualización en pantallas de ordenador y posteriormente se adaptan los diferentes módulos para que se visualicen correctamente en todo tipo de dispositivos ya sea de tamaño pequeño (como smartphones) o de tamaño mas grandes (como pueden ser las smartTV).

[13] *Responsive web Design:* filosofía de diseño y desarrollo web cuyo objetivo es adaptar la apariencia de los sitios web a cualquier dispositivo en el que se visualice.

Algunas de las aplicaciones que están programadas como aplicación web son las siguientes:

|  |  |  |
|---|---|---|
| **GOOGLE DOCS** | **ALIEXPRESS** | **CODEPEN.IO** |
| Año de creación: 2007 | Año de creación: 2010 | Año de creación: 2012 |

Actualmente existen empresas que han creado un producto que facilita la adaptación de páginas web a aplicaciones móviles, un ejemplo de ellas es Reskyt[14].

## *2.2.3 Aplicaciones Híbridas*

Las aplicaciones híbridas o Cross-platforms son aplicaciones que combinan aspectos de las dos anteriores para sacar lo mejor de cada una según los requisitos necesarios. Nuestros dispositivos renderizan una aplicación web en forma de aplicación nativa.

Son un tipo de aplicaciones que se programan mediante los estándares web Javascript, HTML y CSS igual que las Web apps, pero a su vez pueden utilizar las funcionalidades (GPS, sensores, cámara…) del dispositivo dónde se alojan, como en el caso de las aplicaciones nativas.

Son aplicaciones que reducen el coste de desarrollo de éstas últimas, pero a la vez mejoran la experiencia del usuario de una aplicación web ya que están más optimizadas y aprovechan los recursos del dispositivo dónde se ejecutan por lo que, por ejemplo, la navegación es más óptima. También es importante destacar que, si lo comparamos con el rendimiento de las aplicaciones nativas, es menor puesto que éstas aprovechan más óptimamente los recursos de hardware.

Se alojan en el dispositivo por lo que, igual que en las nativas, pueden funcionar sin acceso a internet (siempre y cuando la aplicación no lo necesite obligatoriamente). Son aplicaciones que podemos encontrar en las App Store de cualquier dispositivo.

Muchas de las aplicaciones mas descargadas de Google Play, y que la mayor parte de usuarios usamos a diario son aplicaciones híbridas:

|  |  |  |
|---|---|---|
| **GMAIL** | **FACEBOOK** | **CODEPEN.IO** |
| Año de creación: 2005 | Año de creación: 2007 | Año de creación: 2008 |
|  |  |  |
| **UBER** | **INSTAGRAM** | **PRINTEREST** |
| Año de creación: 2009 | Año de creación: 2010 | Año de creación: 2012 |

[14] *Reskyt:* Empresa leridana que tiene una plataforma tecnológica que permite crear una aplicación de comercio electrónico de forma muy ágil y sin la necesidad de llevar a cabo una programación extra.

Son multiplataforma, es decir, sólo es necesario programar una vez. Para adaptar los componentes a los diferentes dispositivos y lenguajes se utilizan *frameworks,* los 10 mejores actualmente[15] son:

- → React Native
- → Ionic
- → Flutter
- → Xamarin
- → PhoneGap
- → Corona SDK
- → JQueryMobile
- → Intel XDK
- → Mobile Angular UI
- → Framework 7

Para este proyecto, y siguiendo simplemente un interés personal, se ha propuesto el estudio de los siguientes:

- → **React Native.** Desarrollado por *Facebook*.
- → **Flutter.** Desarrollado por *Google.*

## Flutter


Flutter es un framework desarrollado por Google que permite crear aplicaciones nativas para Android e iOS.

Es de código abierto.

Su primera versión fue lanzada en 2015.

Tiene 3 pilares básicos:

- → **Velocidad**: Muy importante la velocidad del desarrollo, para ello cuenta con *Hot Reload*, que permite visualizar los cambios de la aplicación instantáneamente sin tener la necesidad de detener su ejecución.
- → **Diseños flexibles**: Tiene módulos personalizables dentro de una arquitectura basada en capas que incluye animaciones y efectos que facilitan la experiencia de usuario y proporciona expresión a los diseños de las aplicaciones.
- → **Comportamiento nativo**: sus módulos incluyen las grandes diferencias de comportamiento entre las aplicaciones de Android e iOS (como puede ser el *scrolling*[16], la navegación entre las diferentes pantallas, los botones, los iconos…) este echo genera un comportamiento prácticamente nativo en este tipo de aplicaciones.

Un ejemplo claro de aplicación desarrollada con *Flutter* es **Google Ads**.

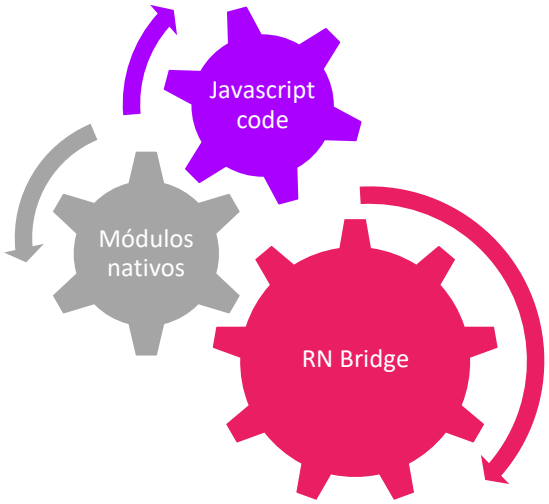
[15] *10 mejores frameworks*: Fuente: Hackernoon, Mattelio, GlowID

[16] *Scrolling:* Es echo de desplazar la ventana que muestra una aplicación para visualizar el contenido completo de ésta.

## React Native

React Native es un *framework* desarrollado por Facebook en 2015, que permite crear aplicaciones nativas mediante javascript. Permite acceder a las APIs y *views* nativas de los dispositivos.

Trabaja con componentes propios que tienen una equivalencia en los diferentes lenguajes de Android e iOS.




Estos son los pilares bàsicos de React Native:

**Mòdulos nativos:** Mòdulos de React Native necesarios de la aplicación.

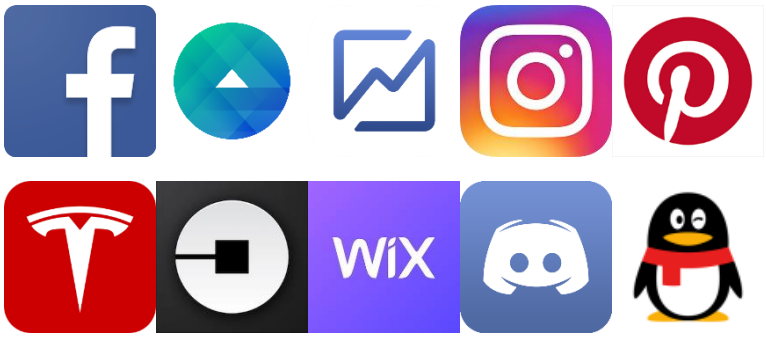
**React Native Bridge:** puente que comunica los diferentes hilos mediante un protocolo de comunicación (basado en Jaca y C++).

**Máquina Javascript Virtual:** Máquina virtual que ejecutara el código de la aplicación.

De igual modo que sucede con *Flutter*, tiene la capacidad de la actualización rápida, característica que facilita la programación ya que permite al usuario visualizar los cambios de forma automática tanto en el emulador como en el dispositivo conectado.

Recibe contribuciones de usuarios individuales y compañías globales como por ejemplo Expo (*framework* que proporciona diferentes componentes a React Native, es uno de los elementos básicos en la aplicación que se ha desarrollado), Infinite Red, Microsoft…

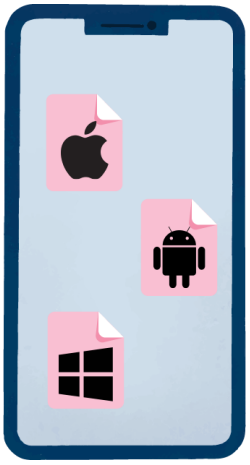
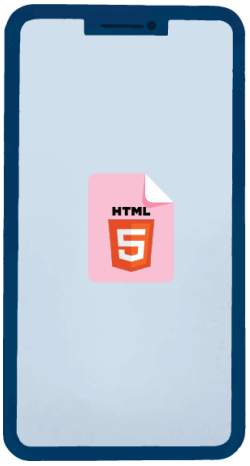
Algunos ejemplos de aplicaciones desarrolladas con React: Facebook y sus diferentes aplicaciones (Facebook analytics, Facebook engineering…), Instagram, Printerest, Tesla, Uber, WiX, Discord, Tencent…).




*Fuente: React Native*

### *2.1.4 Resumen*

Una vez analizadas las diferentes opciones disponibles, se ha realizado un pequeño resumen con las características mas importantes de cada tipo de aplicación para poder tomar la decisión final.

| | |
|---|---|
| **APLICACIONES NATIVAS**<br>  | ✓ No hay necesidad de conexión.<br>✓ Aprovechan las funcionalidades del dispositivo.<br>✓ Visibles en las App Store.<br>✓ Ejecución en 2º plano y notificaciones *push.*<br>✓ Mejor experiencia de usuario |
| | ✗ Desarrollo más costoso.<br>✗ El usuario debe actualizar<br>✗ Código no reutilizable |
| **APLICACIONES WEB**<br>  | ✓ Desarrollo más económico.<br>✓ Código reutilizable.<br>✓ No ocupan memoria.<br>✓ No necesita actualización. |
| | ✗ Necesita conexión.<br>✗ No se pueden usar en 2º plano.<br>✗ No pueden utilizar las características del dispositivo. |
| **APLICACIONES HÍBRIDAS**<br>  | ✓ Coste más económico que las aplicaciones nativas.<br>✓ Multiplataforma.<br>✓ Diferentes *frameworks* en el mercado.<br>✓ Mantenimiento menos costoso. |
| | ✗ No acceden a todas las prestaciones del dispositivo.<br>✗ Requieren de conexión (generalmente).<br>✗ Menos rendimiento. |

## 2.2 Análisis de la competencia

Nos encontramos en una sociedad tecnológica dónde prácticamente todo lo que se pueda pensar está inventado. Realizar una aplicación innovadora y rompedora es difícil, puesto que actualmente existen aplicaciones para casi cualquier tarea, por ese motivo es imprescindible realizar un análisis del mercado.

Ver qué tipo de aplicaciones están en las App Store al alcance de los usuarios puede resultar muy útil. Analizar la posible competencia, sus puntos fuertes y/o debilidades, puede ayudar a alimentar los requisitos principales de la aplicación e incluso descartar algunos detalles que quizá no sean tan útiles o necesarios. También es de mucha ayuda leer los comentarios que los usuarios han dejado sobre estas aplicaciones, ver qué les ha gustado, que echan en falta etc.…

Cogiendo el objetivo del proyecto como base: disponer una plataforma que muestre la información de ocio de la zona, se ha realizado un trabajo de investigación de aplicaciones parecidas que cumplen con todos o varios de los requisitos básicos del proyecto, que se mostrarán más adelante. Se han analizado las más relevantes cogiendo como punto de referencia diferentes características:

- → Estado de desarrollo.
- → Plataformas en las que se encuentra
- → Público objetivo de la aplicación
- → Tareas que los usuarios pueden hacer.
- → Alcance geográfico

Siguiendo estas pautas, se ha realizado un estudio buscando tanto en internet como en las tiendas de aplicaciones de eventos o de ocio. Y se han escogido las más valoradas tanto a nivel mundial como las existentes en el mercado español. Una vez revisadas los resultados de las búsquedas, las aplicaciones más parecidas que se han encontrado en el mercado son las siguientes:

## 2.2.1 EVENTBRITE

| | |
|---|---|
| **Sitio web:**<br>www.eventbrite.es<br><br>**Año de lanzamiento**<br>2006<br><br>**Sede:**<br>San Francisco, Reino Unido, Alemania y Brasil.<br><br>**Tiene aplicación móvil**<br>(Más de 10 M de descargas en GooglePlay)<br><br>**Su alcance geográfico es a nivel mundial.** | **Eventbrite** es una plataforma web dirigida a la venta de entradas y promoción de eventos. Está más dirigida a la organización de eventos puesto que aporta una gestión de las compras de los usuarios de la aplicación y sitio web y su objetivo principal es aumentar las ventas de sus usuarios (los organizadores).<br>El usuario puede acceder a todo tipo de eventos y recibir la entrada por correo electrónico.<br><br>El organizador tiene la posibilidad de crear un sitio web con toda la información del evento (cómo podría ser la creación de un evento en Facebook), se esta forma todos los usuarios pueden ver el evento y tienen la posibilidad de comprar sus entradas. La plataforma ofrece al organizador todo un informe de alcance y ventas de cada evento.<br><br>También ofrece la posibilidad de integrarse con diferentes redes sociales, así como utilizar el móvil como entradas usando QR.<br><br>Esta aplicación promociona todo tipo de eventos (familiares, lúdicos, culturales…). |

## 2.2.2 FOURSQUARE

| | |
|---|---|
| **Sitio web:**<br>www.foursquare.com<br><br>**Año de lanzamiento**<br>2009<br><br>**Sede:**<br>San Francisco, Londres, Nueva York.<br><br>**Tiene aplicación móvil**<br>(Más de 10 M de descargas en GooglePlay)<br><br>**Su alcance geográfico es a nivel mundial.** | **Foursquare** es una red social basada en la localización de locales, restaurantes… mediante la información que extrae de las diferentes redes sociales (Twitter, Facebook, Google Maps…).<br><br>El usuario puede encontrar, marcar y recomendar lugares, crear listas de favoritos y conseguir insignias y puntuaciones a medida que realiza estas acciones, cuántas más interacciones más puntuación.<br><br>La aplicación cuenta con un chat para permitir a los usuarios hablar entre ellos.<br><br>Es una red social que ha conseguido de alguna manera digitalizar el concepto de páginas amarillas.<br><br>No ofrece la posibilidad de comprar entradas, es más una aplicación informativa. Si ofrece diferentes subproductos: *Places, Analytics* etc. Que ayudan al cliente a potenciar su lugar y a ver cuántas interacciones ha conseguido.<br><br>Esta aplicación promociona todo tipo de eventos (familiares, lúdicos, culturales…). |

### 2.2.3 FEVER

| | |
|---|---|
| **Sitio web:**<br>www.feverup.com<br><br>**Año de lanzamiento**<br>2012<br><br>**Sede:**<br>Madrid, Londres, Nueva York.<br><br>**Tiene aplicación móvil**<br>(Más de 1 M de descargas en GooglePlay)<br><br>**Su alcance geográfico abarca diferentes ciudades concretas.** | **Fever** es una plataforma web y aplicación dónde encontrar qué te ofrece una ciudad. Tanto des del sitio web cómo des de la aplicación para Android e iOS, ofrece una lista de eventos que se realizan en la ciudad deseada.<br><br>Para mostrar estos eventos utiliza la información del gráfico social del usuario (amigos, recomendaciones, ubicación etc…) para sugerirle planes que puedan gustarle.<br><br>El usuario no puede adquirir las entradas, pero sí puede encontrar descuentos a partir de la aplicación.<br><br>Esta aplicación promociona todo tipo de eventos (familiares, lúdicos, culturales…).<br><br>Actualmente ofrece promoción en ciudades concretas: Nueva York, Londres, Manchester, Paris, Los Angeles, Lisboa, Barcelona, Madrid, Ibiza, Valencia, Sevilla, Málaga y Bilbao. |

### 2.2.4 FIESTIFY

| | |
|---|---|
| **Sitio web:**<br>Actualmente no dispone de sitio web.<br><br>**Año de lanzamiento**<br>2017<br><br>**Sede:**<br>Madrid<br><br>**Tiene aplicación móvil**<br>(Sin usuarios ni opiniones) | **Fiestify** es una aplicación y plataforma web para todos los públicos y dónde aparece todo tipo de eventos (des de conciertos, parques temáticos, discotecas, teatros…) de Madrid, Barcelona y Valencia.<br><br>El usuario puede buscar el evento que más desee y hacer recomendaciones.<br><br>Al inicio de este proyecto esta aplicación tenia disponible una página web, pero en el momento de la redacción de esta memoria no existe dicho sitio web y la aplicación móvil no se puede encontrar fácilmente en Google Play, y no dispone ni de usuarios ni de descargas. |

## 2.2.5 2NAIT

| | |
|---|---|
| **Sitio web:**<br>Actualmente no dispone de sitio web.<br><br>**Año de lanzamiento**<br>2017<br><br>**Sede:**<br>Sevilla<br><br>**Tiene aplicación móvil**<br>(Más de mil de descargas en GooglePlay)<br><br>**Su alcance geográfico es Sevilla.** | **2Nait** es una aplicación móvil que sirve se guía de locales (muestra sus horarios, ofertas y el ambiente que hay) de Sevilla.<br><br>También actúa de relaciones públicas puesto que permite la venta de entradas anticipadas.<br><br>El usuario puede consultar los eventos de las noches Sevillanas y comprar, si lo desea, su entrada anticipada para ahorrarse las colas de los locales. |

## 2.2.6 PARTYADVISOR

| | |
|---|---|
| **Sitio web:**<br>www.partyadvisorapp.com<br><br>**Año de lanzamiento**<br>2018<br><br>**Sede:**<br>Barcelona<br><br>**Tiene aplicación móvil**<br>(Más de 5.000 de descargas en GooglePlay)<br><br>**Su alcance geográfico abarca diferentes capitales españolas.** | **PartyAdvisor** es la plataforma más parecida a este proyecto. Muestra la información del ocio nocturno de las diferentes ciudades.<br><br>El usuario puede encontrar los eventos por tipo de música, edad, localización y calendario. Puede adquirir entradas, e incluso apuntarse a la lista de invitados.<br><br>Al inicio de este proyecto tenían una aplicación en desarrollo y solo estaban en Madrid, Barcelona, Valencia y Málaga, en el momento de la redacción de esta memoria dicha aplicación ya está disponible en las diferentes plataformas y su alcance también ha augmentado gracias a una ronda de financiación. |

### 2.2.7 CONCLUSIONES DEL ESTUDIO

El mercado de los eventos es muy amplio, y hay mucha competencia. Se han encontrado diferentes aplicaciones, blogs y plataformas web que analizan locales, eventos y ocio en general de las diferentes ciudades.

Como se ha analizado, cada aplicación tiene sus puntos fuertes y sus puntos débiles, por suerte las aplicaciones móviles permiten un amplio abanico de posibilidades de desarrollo que pueden hacer de una aplicación un éxito o un fracaso.

Considerando los siguientes puntos:

- → Eventos focalizados en el **ocio nocturno**.
- → Geolocalización de los **locales de ocio cercanos**.
- → **Compra de entradas** en la misma aplicación, con lista de favoritos y carrito (para dar la opción al usuario a comprar las entradas más adelante).
- → **Validación de la entrada mediante QR** disponible en la misma aplicación para hacer mucho más fácil su compra y el acceso a los locales.
- → **Código *cross-platform*[17]** para que se pueda descargar en los diferentes dispositivos.

Se puede considerar que la aplicación que se quiere desarrollar en este proyecto es un potente competidor de las anteriores puesto que reúne muchos aspectos positivos de cada una de ellas.

Por una parte, focalizar la oferta de eventos a los relacionados con el ocio nocturno, permite reducir el público objetivo y llegar a él con más facilidad. Por otra, el desarrollo de una aplicación *cross-platform* también facilita que usuarios de diferentes sistemas operativos accedan a nuestra aplicación, echo que no es posible en alguna de las propuestas anteriores puesto que han apostado por desarrollo nativo en un lenguaje (en IOS o Android) o apostando por plataforma web.

El diseño atractivo y adaptable de la aplicación también es un punto a favor puesto que se ha tenido muy en cuenta las guías de diseño de *Material Design*[18] de Google (adaptando los diferentes componentes a iOS). Este punto no es diferenciador de las otras aplicaciones, pero es una característica que nos hace estar en las tendencias de mercado.

## 2.3 Framework elegido para el proyecto

Una vez realizado el análisis de las posibilidades que ofrece el desarrollo de aplicaciones móviles y haber hecho un estudio de la posible competencia, sus métodos y su trabajo, se decidió por realizar la aplicación híbrida con **REACT NATIVE**.

Cómo se ha podido ver en el estudio, las aplicaciones híbridas permiten aprovechar muchos de los puntos fuertes y positivos tanto de las aplicaciones nativas cómo de las web apps. Un desarrollo más económico y ágil como las aplicaciones web (ya que React Native se basa en

[17] *Cross-platform:* programas informáticos, aplicaciones, métodos o conceptos de cómputo que se implementan e interoperan en diferentes plataformas informáticas.

[18] *Material Design:* Normativa de diseño enfocado a la visualización del sistema operativo Android, tanto en web cómo en cualquier plataforma. Es una normativa desarrollada por Google y anunciada en el 2014.

Javascript, HTML y CSS) pero que a la vez permite acceder a las prestaciones del dispositivo en el que se aloja (sensores, cámara, GPS….).

El echo de que gigantes de la tecnología y de las aplicaciones como Facebook o Google apuesten por este tipo de desarrollo, es un importante indicador del camino que está tomando el desarrollo de aplicaciones móviles.

Escoger un *framework* desconocido como React Native ha supuesto un reto, tanto personal como para el curso del proyecto puesto que se debe añadir a la lista de tareas programadas el aprendizaje del funcionamiento de este *framework*. Aun así, es un hecho que el mercado laboral actual está demandando desarrolladores que conozcan y hayan trabajado con este lenguaje, y eso ha sido una motivación para aprenderlo y aplicarlo en un proyecto personal.

React Native ofrece una solución a todos los requisitos del proyecto y facilita la adaptación a las diferentes plataformas. Llegar a más usuarios de los que se podría llegar con aplicaciones nativas (en el mismo período de tiempo) es un punto a favor, como ya se ha dicho, solo se necesita programar una vez, a diferencia de hacerlo en nativo que se debería haber hecho con 2 lenguajes lo que supone 2 programaciones diferentes.

En el diseño de la aplicación, se ha querido lograr una visualización ágil y agradable para el usuario, de esta forma se intentará lograr un mayor impacto, y así conseguir una mayor cantidad de futuras descargas en el momento en el que la aplicación se encuentre disponible en las diferentes tiendas de aplicaciones.

Cómo ya se ha comentado anteriormente, el aprendizaje ha supuesto un cambio en el planteamiento inicial de la lista de tareas a realizar. Se ha contemplado la realización de dos cursos a través de la plataforma de cursos digital *Udemy*[19]:

- → Uno para aprender los fundamentos: React, Redux y ECMA Script 6.
- → Otro para aplicar estos conocimientos al desarrollo de la aplicación.

Gracias a estos cursos se ha podido obtener los conocimientos necesarios para realizar el desarrollo de la aplicación que se explicará en el bloque siguiente.

En las diferentes tareas del proyecto también se han utilizado diferentes métodos, plataformas y programas que se irán comentando en cada una.

[19] *Udemy:* Plataforma fundada en 2009 de aprendizaje en línea mediante cursos asequibles impartidos por profesionales que proporcionan herramientas, técnica y conocimientos para formar al usuario en un tema concreto.

# 3. Desarrollo del proyecto

## 3.1 Requerimientos del proyecto

Para elaborar el listado de necesidades del cliente, se mantuvo una primera reunión con él para precisar las necesidades de la aplicación y sus prioridades. De esa reunión salieron los siguientes puntos:

### *3.1.1 Requerimientos funcionales*

R1. El sistema deberá mostrar el listado de eventos más cercanos al usuario según el rango de kilómetros que éste haya establecido (por defecto 120km).
- R1.1. Deben estar ordenados por fecha.
- R1.2. Debe aparecer el cartel del evento.
- R1.3. Debe aparecer el local.
- R1.4. Debe aparecer la distancia a la que se encuentra del usuario.
- R1.5. El usuario tiene que poder buscar el evento que necesite.

R2. El sistema deberá mostrar el listado de locales más cercanos al usuario según el rango de kilómetros que éste haya establecido (por defecto 120km).
- R2.1. Debe estar ordenado por proximidad.
- R2.2. Debe aparecer el logotipo del local (si el promotor lo ha definido).
- R2.3. El usuario tiene que poder buscar la discoteca que necesita.

R3. El sistema tiene que tener una pantalla para mostrar toda la información del evento cuando el usuario lo haya escogido.
- R3.1. Debe aparecer el cartel del evento.
- R3.2. Debe mostrarse el título del evento.
- R3.3. Debe aparecer la fecha y la hora del evento.
- R3.4. Debe aparecer el nombre del local dónde se realiza el evento.
- R3.5. Debe aparecer una breve descripción (si el promotor la ha definido).
- R3.6. Debe aparecer los tipos de música (si el promotor los ha establecido).

R4. En la pantalla del evento el sistema debe mostrar un botón para poder acceder al listado de entradas disponibles para ese evento.

R5. El sistema debe mostrar los tipos de entradas disponibles para cada evento.
- R5.1. Se debe mostrar claramente el tipo de entrada y el precio que cuesta.

R6. El sistema tiene que tener una pantalla para mostrar toda la información del local cuando el usuario lo haya seleccionado.
- R6.1. Debe aparecer el logotipo del local (si el promotor lo ha definido).
- R6.2. Deben aparecer los horarios del local.
- R6.3. Deben aparecer los tipos o tipo de música del local.
- R6.4. Debe aparecer una descripción del local (si el promotor lo ha definido).

R7. El usuario debe poder añadir los locales de ocio que desee en su lista de favoritos.
- R7.1. El sistema tiene que tener una pantalla para mostrar la lista de los favoritos del usuario.
- R7.2. Dicha pantalla debe tener una apariencia parecida a la pantalla que muestra la lista de locales.
- R7.3. El usuario en cualquier momento puede eliminar cualquier ítem de la lista de favoritos.

R8. El usuario debe poder filtrar el listado de eventos.
- R8.1. El sistema debe facilitar el filtro por tipo de música.

R9. El usuario debe poder filtrar el listado de locales.

R9.1. El sistema debe facilitar el filtro por tipo de música.

R10. El sistema debe facilitar al usuario la opción de navegar (con el navegador del dispositivo) al evento dentro de la pantalla de la información del evento.

R11. El sistema debe facilitar al usuario la opción de navegar al local dentro de la pantalla de la información del local.

R12. El usuario debe poder compartir en sus redes sociales la información del evento.

R12.1. Si el sistema no permite escoger cualquier medio para compartir la información, éste deberá dar la opción al usuario para compartirlo en las redes sociales:

R12.1.1. Facebook

R12.1.2. Instagram

R12.1.3. Twitter

R12.1.4. Whatsapp

R12.1.5. Telegram

R13. El usuario debe poder compartir en sus redes sociales la información del local.

R13.1. Si el sistema no permite escoger cualquier medio para compartir la información, éste deberá dar la opción al usuario para compartirlo en las redes sociales:

R13.1.1. Facebook

R13.1.2. Instagram

R13.1.3. Twitter

R13.1.4. Whatsapp

R13.1.5. Telegram

R14. Cuando el usuario escoja el tipo de entrada que desea comprar, el sistema debe mostrar un resumen de compra para que el usuario revise los datos.

R15. Cuando el usuario escoja el tipo de entrada que desea comprar, y haya revisado los datos, el sistema facilitará al usuario un formulario para introducir los datos de la tarjeta de crédito.

R16. Cuando la compra se haya realizado satisfactoriamente, el sistema generará un QR para que el usuario lo tenga disponible cuando llegue al local.

R17. El sistema deberá mostrar un listado con las entradas adquiridas por el usuario:

R17.1. Se deberán mostrar tanto las entradas de los eventos vigentes como las entradas de los eventos pasados.

R17.2. Las entradas de los eventos pasados deberán tener un aspecto visual diferente para no confundir al usuario.

R17.3. Las entradas de los eventos vigentes deberán facilitar la visualización del QR con la compra para que el usuario pueda acceder a él para entrar en el local.

Después de hacer el estudio de mercado, generar un *wireframe* y el prototipo de la aplicación, se volvió a concretar una reunión con el promotor de la cual surgieron más requerimientos funcionales de la aplicación:

R18. El sistema debe agregar (si el promotor lo especifica) un tipo de entrada “Lista”, en la cual deberá facilitar al usuario un formulario de inscripción.

R18.1. Este tipo de entrada no debe conducir a un *checkout*[20] de pago.

R19. El sistema debe facilitar al usuario guardar las tarjetas de pago en la pantalla de *checkout*.

[20] *Checkout:* Proceso de compra, dónde el usuario introduce sus datos, puede visualizar los productos que está adquiriendo y, si el comercio lo requiere, solicitar un tipo de envío, factura…

R19.1. El sistema facilitará una pantalla para que el usuario pueda escoger la tarjeta con la que desea efectuar la compra.

R19.2. El sistema facilitará una pantalla dónde el usuario podrá visualizar y eliminar sus tarjetas favoritas.

R19.3. En ningún momento el sistema deberá mostrar toda la información de la tarjeta, en los momentos en que se visualice un listado de tarjetas SOLO deberán aparecer los últimos dígitos de cada una de ellas.

R20. El sistema deberá facilitar, en la pantalla de 'resumen de la compra', que el usuario pueda introducir el número de RRPP[21] para poder reducir el precio del coste de la entrada (según lo haya especificado el promotor).

R21. El usuario deberá poder guardar el evento en el calendario de su dispositivo.

R22. El usuario deberá poder hacer *Log In* con Facebook.

R23. En el momento en que el usuario entre en la aplicación, ésta deberá enseñar de forma fácil, rápida y concisa la forma de trabajar de Partyfy.

R24. En el caso de integrar en la aplicación un sistema de transferencia de dinero por el móvil*, el sistema deberá facilitar la opción de pedir dinero a otros usuarios.

* El sistema de transferencia de dinero no ha sido un requerimiento necesario en el desarrollo de la aplicación, pero se ha contemplado la posibilidad en el diseño de ésta.

### *3.1.2 Requerimientos no funcionales*

1. El sistema debe desarrollarse en *cross-platform* para facilitar las diferentes opciones al usuario, y así llegar a un mayor número de usuarios.
2. El sistema deberá ser completamente seguro durante el proceso de compra.
3. El sistema NO facilitará la información de las tarjetas del usuario.
4. El sistema informará en todo momento, de la forma más entendible para el usuario, de cualquier error en la aplicación.
   - 4.1. Si hay algún error interno.
   - 4.2. Si el evento / local no existe o no está disponible.
   - 4.3. Si no existen eventos / locales en el radio definido por el usuario.
   - 4.4. Si no hay entradas disponibles en el evento seleccionado.
   - 4.5. Si ha habido algún error durante el envío de cualquier formulario.
   - 4.6. Si hay algún problema con las tarjetas de crédito.
   - 4.7. Cualquier otro problema, error o aviso no contemplado anteriormente.
5. El sistema deberá estar disponible en las tiendas de aplicaciones.
6. El sistema deberá conectarse con un *backend* para acceder a todos los datos de la aplicación, y así cumplir con los requisitos previos expuestos:
   - 6.1. Eventos
   - 6.2. Locales
   - 6.3. Usuarios

[21] *RRPP:* Relaciones púbicas, aquellas personas encargadas de la relación entre una organización (en nuestro proyecto, el local) y la comunidad. En nuestro caso, los RRPP son aquellos encargados de la venta de entradas anticipadas de un local.

6.4. Tarjetas
6.5. Favoritos

7. El sistema debe tener acceso a aplicaciones del dispositivo y sensores (p.e. GPS, aplicación de mapas y calendario).

## 3.2 Esquema MoSCoW

| MUST | SHOULD |
|---|---|
| → **R1.** Visualizar el listado de eventos.<br>→ **R2.** Visualizar el listado de locales.<br>→ **R3.** Visualizar la información del evento.<br>→ **R4.** Acceder a los tipos de entradas.<br>→ **R5.** Visualizar los tipos de entradas.<br>→ **R6.** Visualizar la información del local<br>→ **R15.** Visualizar el formulario de *checkout.*<br>→ **R16.** Generar el QR de la entrada.<br>→ **R17.** Visualizar el listado de entradas.<br>→ **R22.** *Log In* con Facebook | → **R8.** Filtro para los eventos.<br>→ **R9.** Filtro para los locales.<br>→ **R14.** Pantalla del resumen de la compra.<br>→ **R17.1.** Visualizar las entradas pasadas y en un aspecto diferente.<br>→ **R18.** Tipo de entrada "Lista".<br>→ **R19.** Guardar tarjetas usadas / favoritas |
| **CAN** | **WON'T** |
| → **R7.** Añadir un local a la lista de favoritos.<br>→ **R10. / R11.** Acceder al GPS para navegar hacia el evento / local.<br>→ **R12.** Compartir en redes el evento.<br>→ **R13.** Compartir en redes el local.<br>→ **R20.** Introducir el número de RRPP.<br>→ **R21.** Guardar el evento en la aplicación del calendario.<br>→ **R23.** Explicación inicial de la aplicación.<br>→ ***** Animación de finalizar compra | → **R24.** Pedir dinero a otros usuarios.<br>→ ***** Integrar la aplicación con algún sistema de transferencia de dinero con el móvil.<br>→ ***** Integrar la aplicación con Paypal. |

## 3.3 Wireframe

Las siguientes imágenes muestran el conjunto de pantallas generadas para formar el *wireframe* de la aplicación.

El objetivo de este *wireframe* ha sido poder explicar de forma muy simple y esquemática, sin entrar en detalles de diseño mas concretos (colores, fondos, interacciones …) el tipo de información que se mostrará en cada tipo de pantalla.

De esta forma, en la primera reunión con el promotor posterior a la toma de requisitos, se pudo presentar una estructura de las pantallas y de allí surgieron más requisitos y otras sugerencias (cómo por ejemplo la explicación inicial de la aplicación antes de registrarse, o la necesidad de mostrar un resumen de la compra antes de introducir la tarjeta de crédito).


*Ilustración 9. Wireframe - Pantalla inicial*


*Ilustración 10. Wireframe - Pantalla Login*

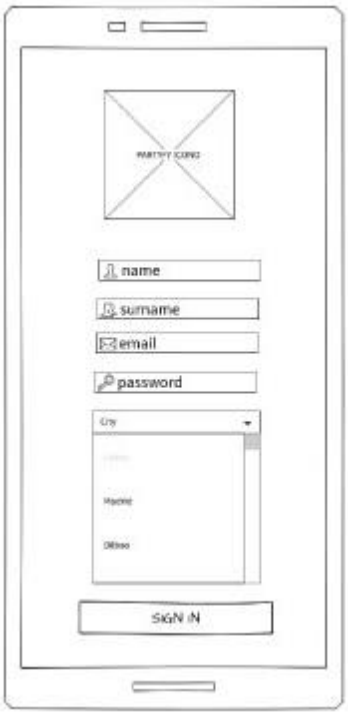



*Ilustración 11. Wireframe - Pantalla Registro*


*Ilustración 12. Wireframe - Pantalla Listado de eventos*


*Ilustración 13. Wireframe - Pantalla Mapa de eventos*

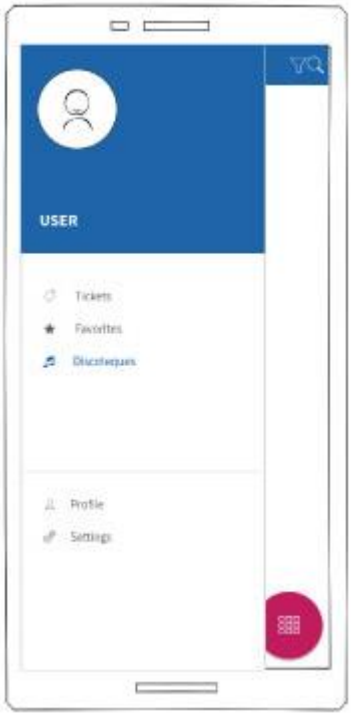



*Ilustración 14. Wireframe - Menu*


*Ilustración 15. Wireframe - Pantalla Listado de tickets*

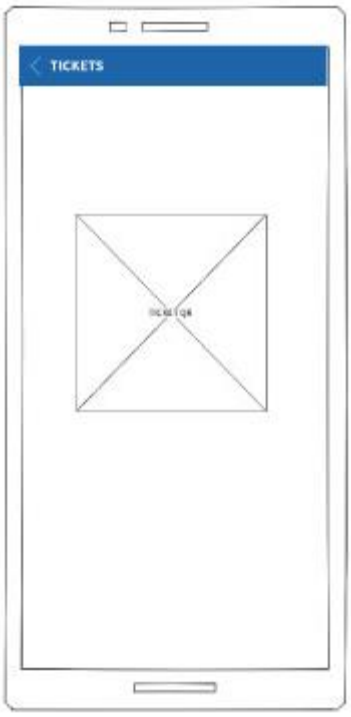



*Ilustración 16. Wireframe - Pantalla QR*


*Ilustración 17. Wireframe - Pantalla configuración*

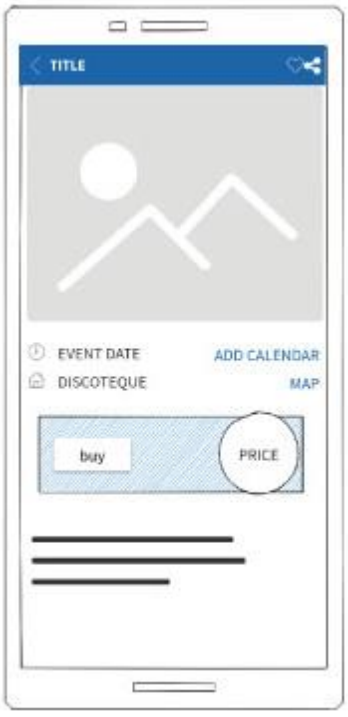



*Ilustración 18. Wireframe - Pantalla Información del evento*


*Ilustración 19. Wireframe - Pantalla tipo de entradas*


*Ilustración 20. Wireframe - Pantalla resumen de compra*

## 3.4 Mockup

Estas son algunos de los diseños de las diferentes pantallas de la aplicación, realizadas con Adobe Illustrator, el listado completo de las pantallas está disponible en el Anexo I.

Estas pantallas ya aparecen con las modificaciones hechas por los usuarios a los que se les hizo el test de diseño una vez terminado este mockup.


*Ilustración 21. Mockup - Pantalla inicial*

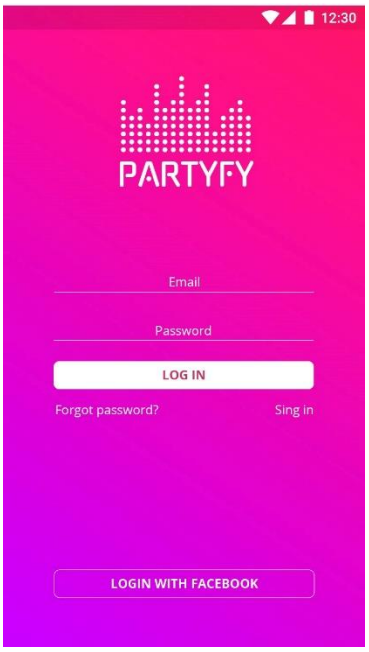



*Ilustración 22. Mockup - Pantalla Login*

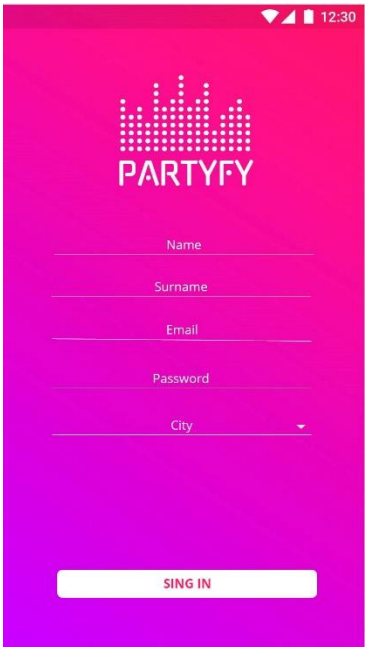



*Ilustración 23. Mockup - Pantalla Registro*

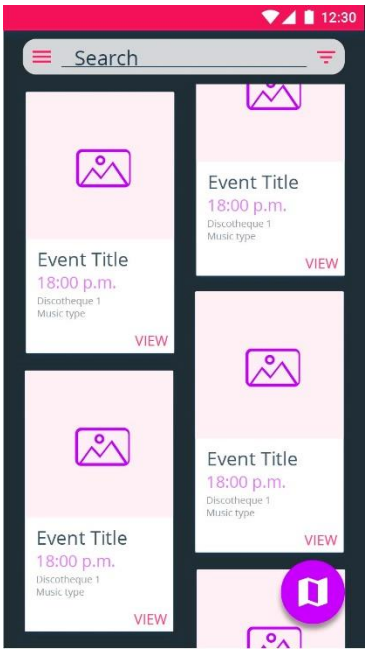



*Ilustración 24. Mockup - Pantalla Listado de eventos*

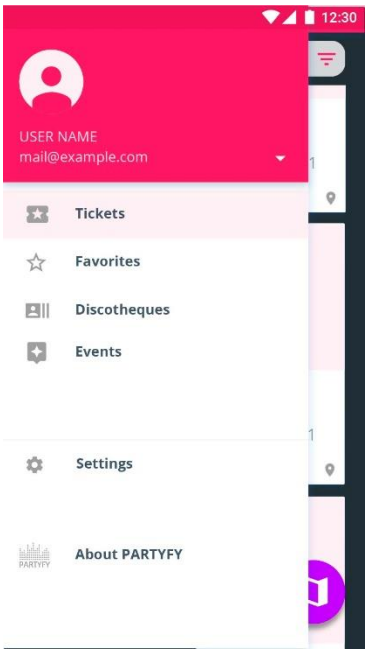



*Ilustración 25. Mockup - Pantalla Menu*

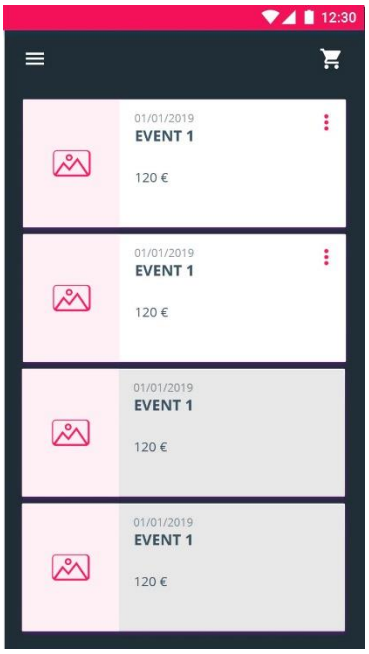



*Ilustración 26. Mockup - Pantalla Listado de entradas*


*Ilustración 27. Mockup - Pantalla QR*

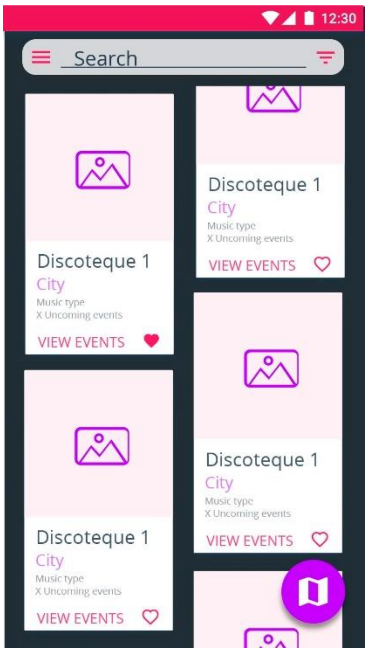



*Ilustración 28. Mockup - Pantalla Listado de locales*

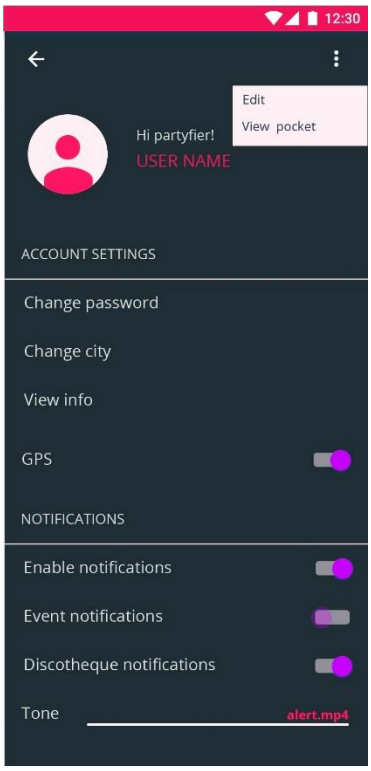



*Ilustración 29. Mockup Pantalla configuración*


*Ilustración 30. Mockup Pantalla información del local*

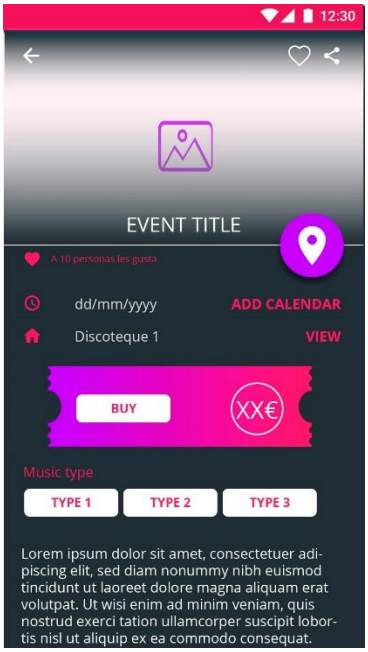



*Ilustración 31. Mockup Pantalla información evento*

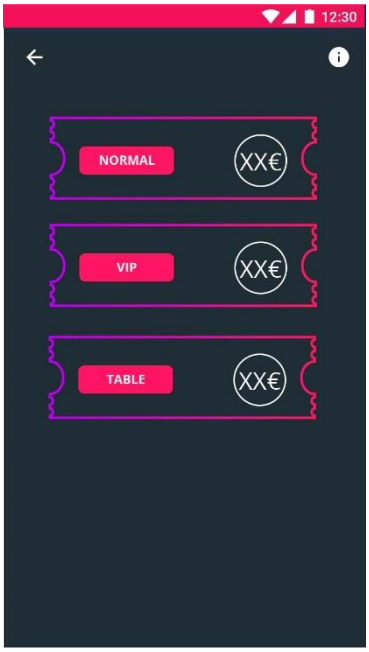



*Ilustración 32. Mockup Pantalla listado de entradas*

Estas son las pantallas que se han usado como base y modelo en el momento de desarrollar la aplicación, en el resultado final ha habido cambios en algunas de ellas por cuestiones técnicas o de diseño. Hay requerimientos que no aparecen en las pantallas (por ejemplo, la explicación de la aplicación o el tipo de eventos “lista”) puesto que son requisitos que se añadieron una vez presentado el mockup al cliente.

## 3.5 Guía de estilo

Una guía de estilo tiene como objetivo recoger los aspectos más importantes de diseño de cualquier proyecto o empresa. En este caso, ha recogido aspectos esenciales de la aplicación para lograr mantener una homogeneidad en los diferentes componentes de las pantallas: botones, fondos, tipografías, presentación del logotipo de la aplicación etc…

De esta forma se asegura que cada ítem descrito en la guía de estilo tiene que seguir las características marcadas por la misma.

### *Logotipo*


### *Paleta de colores*

HEX: #E91E63
RGB: 233, 30, 99

HEX: #AA00FF
RGB: 170, 0, 255

### *Tipografia*

Gugi Regular

Aa Bb Cc 1 2 3

Open Sans Regular

Aa Bb Cc 1 2 3

Open Sans Semibold

**Aa Bb Cc 1 2 3**

### *Imágenes*

*Ilustración 33. Fondo principal de la aplicación*

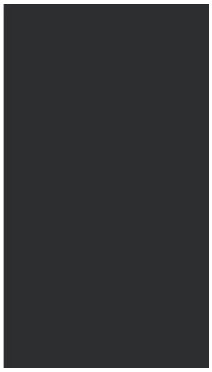


*Ilustración 34. Fondo secundario de la aplicación*

## 3.5 Personas

A continuación, se pretende usar el *Modelo Persona* para mostrar posibles usuarios que podrían usar la aplicación y en las situaciones en las que podría usarla.

**Nombre:** Benito
**Ocupación:** Estudiante
**Personalidad:** Benito es una persona inquieta, muy centrado en sus estudios, pero a la vez le gusta salir con sus amigos. Le gusta buscar muchas posibilidades antes de tomar una decisión.
**Puntos fuertes:** Curiosidad
**Puntos débiles:** No le gusta perder el dinero.

Partyfy podría darle a Benito una solución para los fines de semana que sale con sus compañeros de universidad para desconectar de los estudios. Le ofrece diferentes posibilidades de ocio, punto fuerte para su curiosidad y a la vez le ofrece la opción de comprar las entradas desde la aplicación.

**Nombre:** Lucy
**Ocupación:** Informática
**Personalidad:** Lucy es una persona muy carismática, extrovertida y le gusta mucho cenar con sus amigas el primer miércoles de cada mes.
**Puntos fuertes:** Muy sociable, conoce muchos locales
**Puntos débiles:** No le gusta esperar colas.

En el caso de Lucy, que conoce los locales de ocio de su ciudad, Partyfy podría ayudarla a evitar las colas de las taquillas. Con la aplicación podría comprar las entradas de su local favorito, incluso podría comprar una mesa VIP para ir con sus amigas después de cenar.

**Nombre:** Ainara y Jordi
**Ocupación:** Viajeros
**Personalidad:** Ainara y Jordi son una pareja de recién casados que, de viaje de novios se ha ido a dar la vuelta a España. Son muy divertidos y sociales.
**Puntos fuertes:** Les gusta todo tipo de ambientes.
**Puntos débiles:** No conocen los lugares a donde van.

Para Ainara y Jordi que llegan a ciudades desconocidas, Partyfy les ayudaría a encontrar los locales de ocio. Podrían ver todas las ofertas de la ciudad y escoger la que más les apetezca.

**Nombre:** Esther
**Ocupación:** CEO de una multinacional
**Personalidad:** Esther es una mujer muy inteligente, competente y ocupada, esta continuamente de viaje de trabajo. **Puntos fuertes:** Le gusta bailar salsa en su tiempo libre
**Puntos débiles:** Es muy metódica y le gusta encontrar las cosas rápido.

Esther podrá usar Partyfy cuando le apetezca ir a bailar salsa al local de moda de su ciudad. Podrá filtrarlo fácilmente por el tipo de música e incluso podrá comprar la entrada para no perder el tiempo en el local. También podrá filtrar sus locales favoritos, de salsa o no, para poder encontrarlos la próxima vez fácilmente.

# 3.6 Desarrollo

## *3.6.1 Estructura del programa*

El desarrollo de la aplicación se ha llevado a cabo bajo el concepto de 'programación modular'. Este concepto se refiere a un paradigma de programación que consiste a subdividir el problema en tareas más pequeñas (submódulos o subprogramas) para poder desarrollarlo con más facilidad y que todo sea más ágil y legible.


*Ilustración 35. Capturas del árbol de carpetas.*

Como se puede apreciar en la imagen anterior, la aplicación se ha dividido en diferentes partes:

- → Componentes
- → Navegación
- → Idioma
- → Pantallas
- → *Utils* (otros archivos)
- → *Assets* (Recursos, como son las imágenes, logotipos etc…)

Algunas de las partes de programación se han podido reutilizar en otras pantallas, esas partes son las llamadas **componentes**. Un ejemplo claro es la visualización de la pantalla de eventos o de locales / discotecas; ambas pantallas tienen la misma estructura, lo único que cambia es la información que se muestra, por ese motivo hemos creado el componente *CardItem* dónde se muestra la información del evento o de la discoteca.

Algunos componentes son genéricos, como el caso del *Preloader* (la pantalla que se muestra mientras carga la información), *TopSearchBar* (la barra superior de búsqueda) o *BackgroundImage* (que es uno de los fondos que se han utilizado para la aplicación, correspondiente a un degradado en los tonos del logo).

La aplicación esta preparada para ser traducida de forma ágil, no era un requisito del sistema, pero se ha considerado que es un punto importante para facilitar la traducción a múltiples idiomas y así llegar a más público. Esta traducción se guarda en la carpeta **lang**.

En esta carpeta se guarda un archivo para cada idioma donde se especifican las cadenas que aparecen en la aplicación (botones, formularios, explicación… etc). De esta forma, en la aplicación en lugar de escribir el texto que aparecerá en el componente, se llama a la variable que lo guarda. En la siguiente imagen podemos ver un fragmento del documento del español:

```
export default es = {
    login: 'Login',
    register: 'Registro',
    name: 'Introduce tu nombre',
    lastname: 'Introduce tus apellidos',
    email: 'Introduce tu email',
```

*Ilustración 36. Variables de idioma español.*

Para saber qué pantallas debe mostrar el sistema cuando el usuario realiza las diferentes acciones de la aplicación (entrar por primera vez, *login*, ver un evento, comprar una entrada…) se usan las ***navigations***. Hay dos tipos:

- → ***Guest:*** Dónde se especifica que, si el usuario que entra en la aplicación no está logueado, le muestre la pantalla de bienvenida.
- → ***Logged:*** Este apartado muestra todo el árbol de navegación de cada pantalla una vez el usuario está registrado en la base de datos.

Todas las pantallas de la aplicación se guardan en la carpeta ***screens***. Se ha dividido en subcarpetas para hacer mas fácil la programación y para agrupar los diferentes ficheros .js correspondientes a las pantallas según qué tipo de información muestran (si es una pantalla de eventos, de discotecas, de las ventas, del usuario etc..).

En el apartado ***Utils*** se guardan todos los ficheros de configuración, de conexión a la base de datos, de conexión con el *backend* o también la validación de los formularios.

El apartado de ***Assets*** o recursos, es dónde se guardan todas las imágenes, animaciones, iconos, videos etc… Todos los recursos gráficos que usa la aplicación.

Finalmente, los siguientes archivos son importantes puesto que son la base de todo lo explicado anteriormente:

- **App.js:** Es la raíz de la aplicación, es donde se especifica qué tipo de navegación deberá mostrar el sistema según si el usuario está logueado o no (*logged* o *guest*).
- **Package.json:** Es el archivo donde se declaran todas las dependencias de la aplicación, todas las librerías utilizadas y sus versiones.

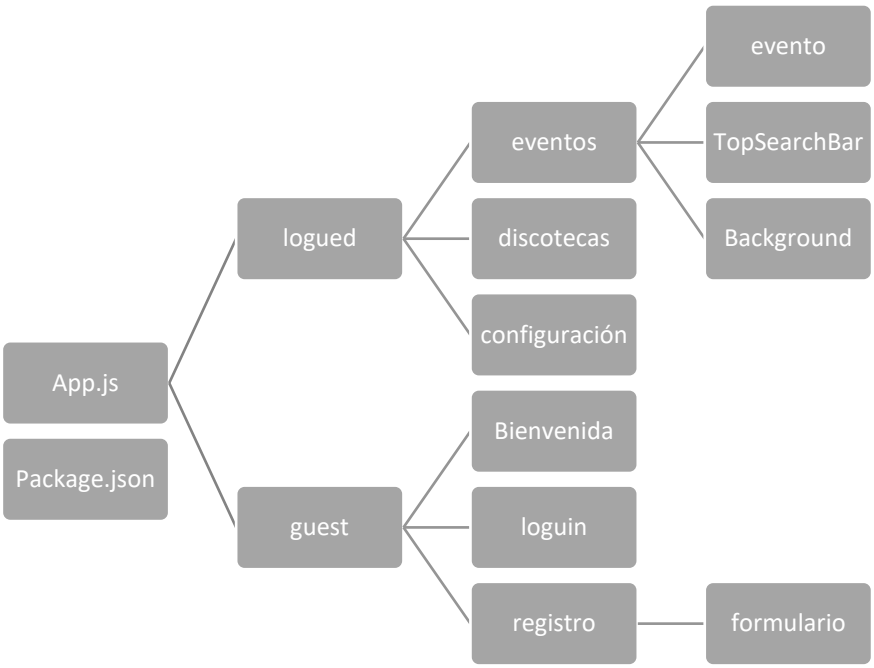



*Ilustración 37. Ejemplo de estructura*

### 3.6.2 Conexión a la base de datos

Como se ha comentado en otras partes del documento, la aplicación obtiene y modifica los datos de la base de datos mediante peticiones al *backend*. Éste consiste en una API-REST[22] conectada a la base de datos que devuelve las respuestas en formato JSON.

Estas respuestas pueden ser satisfactorias, es decir, devuelve el JSON de los datos que pide la aplicación, o erróneas, que en este caso devuelve el código del error. Con el *http status code* [23] se ha gestionado los diferentes resultados de las consultas.

Las respuestas en formato JSON son leídas y tratadas por la aplicación que, una vez dividida la respuesta en las diferentes variables, las trata y las envía a los diferentes componentes para que se visualicen de la forma que tienen que aparecer en la aplicación (por ejemplo, en una *Card,* o como texto de un botón en el caso del tipo de entradas, o como una *Image* en el caso del cartel del evento o logotipo de la discoteca).

Todas estas peticiones a la API están securizadas mediante un *token* de acceso que se obtiene una vez el usuario ha hecho *log in* en la aplicación. Esta información de guarda en la base de datos local de la aplicación mediante un *AsyncStorage*.

### 3.6.3 Problemas resueltos

Durante el desarrollo de la aplicación han aparecido diferentes contratiempos, algunos de ellos y cómo se han solucionado se muestran a continuación:

- **Actualización del SDK de Expo.** Durante la programación de la aplicación se ha tenido que actualizar en diferentes ocasiones los paquetes descritos en el package.json puesto que los desarrolladores de éstos habían hecho cambios o mejoras. En el caso de la actualización del SDK de Expo se ha tenido más complejidad a la hora de llevarlo a cabo porque alguna de las librerías usadas en la aplicación ha cambiado de nombre, así como los métodos en la última versión.
  Al actualizar la aplicación ha dejado de funcionar, y se ha tenido que revisar las notas de la *update* para poder ver los cambios en las nomenclaturas de las librerías y la forma correcta de exportarlas de nuevo.
- **Refrescar los datos.** Una vez leídos los datos de la aplicación, ha surgido el problema de la actualización de la información. Si en la base de datos se añade o se elimina algún evento y/o local, la aplicación en un principio tenía que cerrarse para volver a obtener la información.
  Este error se ha solucionado usando el método *_onRefresh()* que consiste en que, cuando el usuario hace *Swipe* (gesto de deslizar la pantalla hacia abajo) la aplicación vuelve a ejecutar la petición de los datos al *backend* sin necesidad de reiniciarla.

[22] *API-REST:* API es un conjunto de reglas y especificaciones que las aplicaciones pueden seguir para comunicarse entre ellas. Una API-REST es un protocolo cliente-servidor que contiene diferentes operaciones (las mas importantes POST, GET, PUT, y DELETE). Normalmente intercambian datos mediante formatos como JSON o XML.

[23] *Http status code:* Es una lista de códigos de estado de respuesta de HTTP. Es parte del estándar de HTTP/1.1.

### *3.6.4 Test funcional de la aplicación*

En todo el proceso de desarrollo de la aplicación se han podido ver los cambios en directo del código puesto que se ha ejecutado en un dispositivo Android físico y en un emulador de Iphone, utilizando en todo momento las ultimas versiones de cada uno de los sistemas operativos.

A continuación, se pueden observar algunas de las capturas de pantalla que muestran la mayor parte de requisitos de la aplicación.


*Ilustración 38. Primera pantalla de la aplicación y del slider de bienvenida*

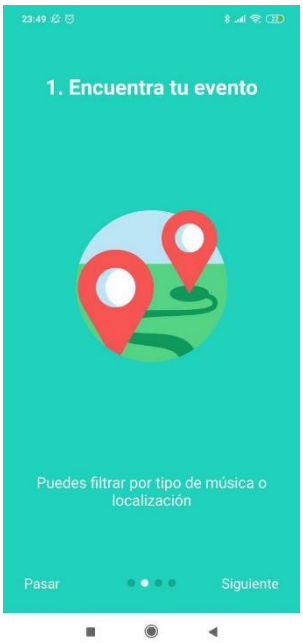



*Ilustración 39. Primer item del slider de la explicación de la aplicación*


*Ilustración 40. Primer item del slider de la explicación de la aplicación*

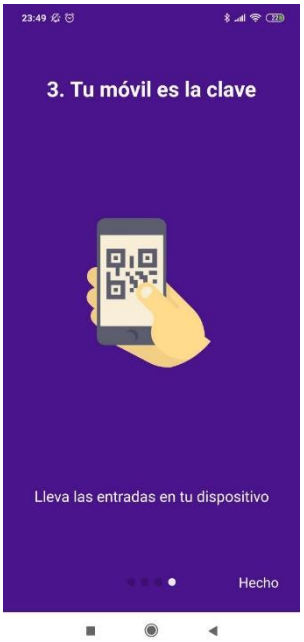



*Ilustración 41. Primer item del slider de la explicación de la aplicación*


*Ilustración 42. Pantalla de Log in*


*Ilustración 43. Pantalla de registro*


*Ilustración 44. Pantalla de la lista de eventos*

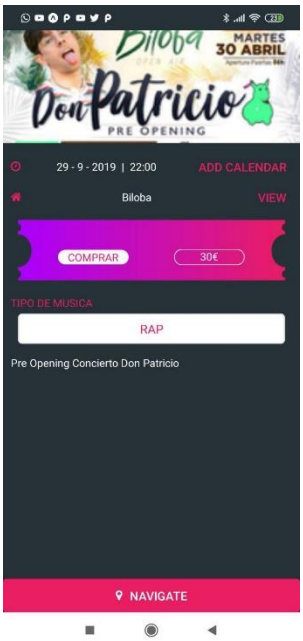



*Ilustración 45. Pantalla de información del evento*

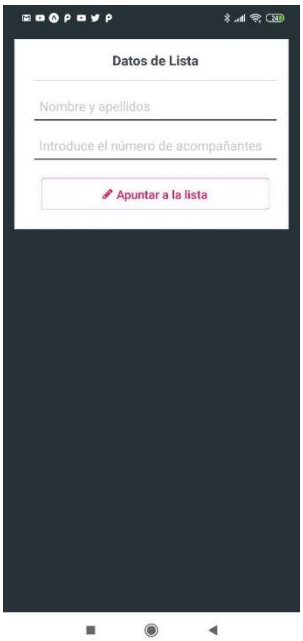



*Ilustración 46. Pantalla de inscripción a la lista de invitados*

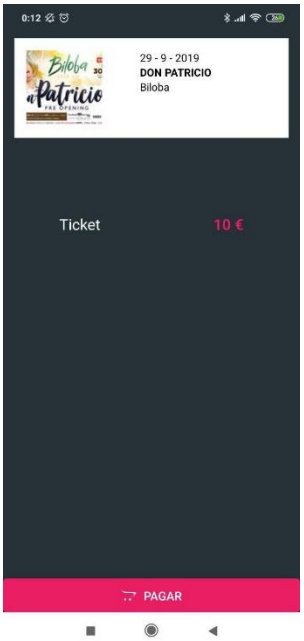



*Ilustración 47. Pantalla de resumen de compra*

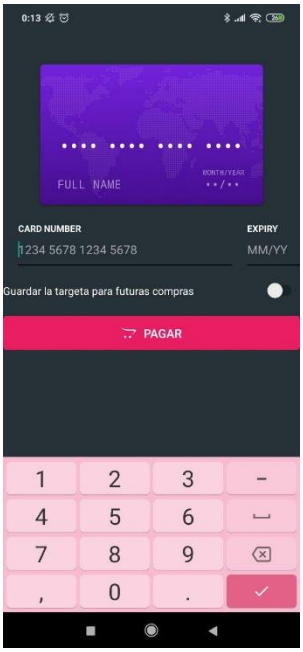



*Ilustración 48. Pantalla para introducir nueva tarjeta*

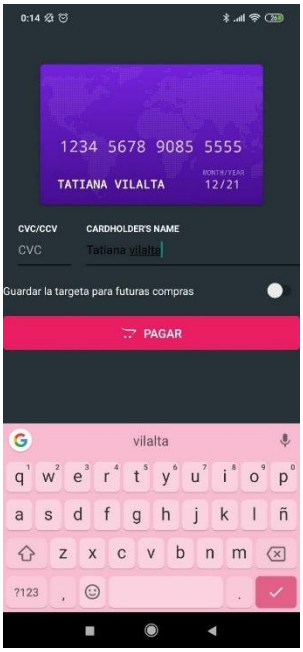



*Ilustración 49. Pantalla para introducir nueva tarjeta (bis)*

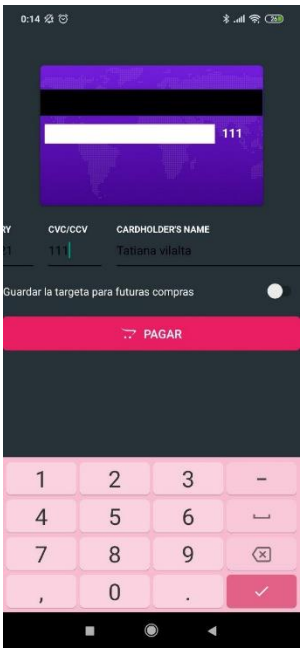



*Ilustración 50. Pantalla para introducir nueva tarjeta (detras)*


*Ilustración 51. Pantalla menu*


*Ilustración 52. Pantalla del listado de locales*


*Ilustración 53. Pantalla de la información del local*


*Ilustración 54. Pantalla de 'Mis entradas'*

# 4. Finalización del proyecto

## 4.1 Conclusiones

Si tomamos el punto de partida el objetivo del proyecto:

> *El objetivo de este trabajo consiste en crear una aplicación móvil que muestre toda la información extraída de un backend externo. Es una información referente a eventos de una ciudad o zona más cercanos al usuario que la tiene instalada en su dispositivo.*

Considero que se ha podido realizar la mayor parte del trabajo que se había planteado. Hay algunos requisitos que no se han podido completar que los detallaremos en el siguiente punto (4.2) pero en general se ha realizado una aplicación funcional, que permite visualizar la información, realizar la compra de entradas y ver los locales de ocio de la zona.

Durante el desarrollo se han tenido diferentes problemas de compatibilidad de algunas librerías, o de actualización de *Expo* (librería base del proyecto) puesto que en 2 ocasiones se han realizado actualizaciones importantes de la versión que han obligado a cambiar algunos métodos.

Se ha podido aprender las bases y fundamentos de una nueva forma de trabajo y de desarrollo de aplicaciones como es la de usar un *framework* (en este caso React Native) que era uno de los objetivos personales más importantes.

Una vez trabajado con ambas opciones (aplicaciones nativas e híbridas) considero que ambas son muy válidas, y que la decisión entre ellas se basa en el tiempo y el presupuesto del proyecto. Una aplicación nativa aprovecha mucho mas las características de los dispositivos y, a nivel de diseño, es más ágil puesto que puedes usar elementos nativos de cada SO (cómo por ejemplo los *Toast* para Android, los avisos en forma de popup) y no te limita a usar elementos que puedan valer para todos. Las aplicaciones hibridas son más ágiles en cuanto a programación se refiere ya que, una vez programada ya se adapta a otras plataformas, por lo tanto, te ahorra tiempo y en este sector el tiempo es dinero.

Para próximas aplicaciones o proyectos me gustaría profundizar más en las bases del diseño centrado en el usuario, en este proyecto no he podido realizar todos los pasos concretos para ello, sino que simplemente me he basado en algunas características y puntos clave. Este es un punto importante puesto que en el mundo de las aplicaciones el elemento principal es el usuario y, si la aplicación es útil y fácil de usar, la probabilidad que muchos usuarios se la descarguen es elevada.

Para ser críticos, considero que me ha faltado un poco de organización en el proyecto ya que combinarlo con las horas de trabajo ha sido complejo. No he podido realizar una aplicación completa (*backend* y *frontend*) por falta de conocimientos, pero como he podido observar en este proyecto, internet es una fuente de conocimiento, cursos, tutoriales y ejemplos inmensa, así que es un propósito para el próximo proyecto.

En resumen, con este proyecto he aprendido conocimientos nuevos, aplicaciones desconocidas y me ha ayudado a organizarme un poco más las tareas a realizar. He aprendido a desarrollar una aplicación real, para un cliente real, con todos los pasos a seguir que eso conlleva. En este proyecto, aparte de aprender, también he podido poner en práctica muchos de los

conocimientos adquiridos durante el grado de ingeniería informática como pueden ser los conocimientos de IPO o la toma de requisitos, por ejemplo.

## 4.2 Trabajo futuro

En la aplicación desarrollada han quedado pendientes de finalizar algunos de los requisitos.

Han quedado pendientes los filtros de la aplicación, así como poder compartir el evento en las redes sociales. Tampoco se han podido documentar las pruebas a usuarios, así como probar la aplicación final.

La visualización de los eventos en el mapa y de los locales tampoco se ha podido finalizar por problemas de desarrollo, puesto que se han priorizado otros requisitos más urgentes.

Algunos de estos requisitos no se han podido finalizar por falta de tiempo y otros por modificaciones hechas en el *backend* que no se han podido aplicar en la aplicación a tiempo para la redacción de esta memoria, aún así la aplicación se finalizará con todos los requisitos del cliente para poder colgarse en las diferentes tiendas de aplicaciones y que pueda ser utilizada por los usuarios.

# 5. Bibliografia

Rafa ARJONILLA. Rafa Arjonilla [online]. Disponible: https://rafarjonilla.com/que-es/backend/

Andrea CANTÚ. Intuitivamente [online]. Disponible: https://blog.acantu.com/que-es-ux-y-ui/

Microsoft. [online]. Disponible: https://support.office.com/es-es/article/presentar-datos-en-un-diagrama-de-gantt-en-excel-f8910ab4-ceda-4521-8207-f0fb34d9e2b6

Portfoli d'Avaluació contínua. Escola Politècnica Superior [online]. Disponible: http://www.eps.udl.cat/export/sites/Eps/docs/secretaria/TFG-TFM/Portafoli_avaluacio_continuada_TFG-TFM.pdf

Color tool. Material.io [online]. Disponible: https://material.io/tools/color/

Material Design. Material.io [online]. Disponible: https://material.io/design/components/

Tipos de aplicacions móviles; natives, webs, híbrides. Solbyte [online]. Disponible: https://www.solbyte.com/blog/2014/07/21/tipos-de-aplicaciones-moviles-nativas-webs-hibridas/

¿App nativa, web o híbrida?. Raona [online]. Disponible: https://www.raona.com/aplicacion-nativa-web-hibrida/

Apps Hibridas vs Nativas vs Generadas. InnovaAge [online]. Disponible: https://www.innovaportal.com/innovaportal/v/696/1/innova.front/apps-hibridas-vs-nativas-vs-generadas-que-decision-tomar

Desarrollando aplicaciones móviles nativas con React Native. Paradigma digital [online]. Disponible: https://www.paradigmadigital.com/dev/desarrollando-aplicaciones-moviles-nativas-con-react-native/

Flutter [online]. Disponible: https://flutter.dev/

Método MoSCoW [online]. Disponible: https://jummp.wordpress.com/2013/04/27/metodo-moscow/

Material UI [online]. Disponible: https://material-ui.com/

React Native App intro Slider. Jacse [online]. Disponible: https://github.com/Jacse/react-native-app-intro-slider#example

Tcomb-form-native. Alvaromb [online]. Disponible: https://github.com/gcanti/tcomb-form-native

Metodología Scrum. Sinnaps [online]. Disponible: https://www.sinnaps.com/blog-gestion-proyectos/metodologia-scrum

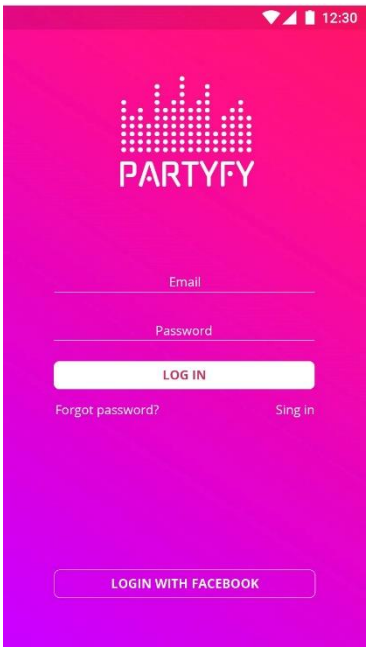
Mobile Operating System Market Share WorldWide. GlobalStats[online]. Disponible: https://gs.statcounter.com/os-market-share/mobile/worldwide/

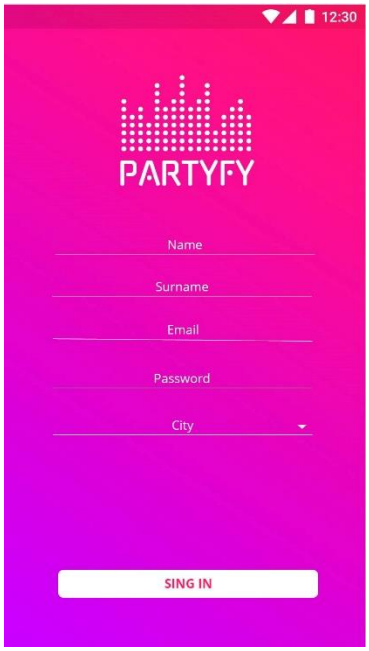
React Native Elements. [online]. Disponible: https://react-native-training.github.io/react-native-elements/

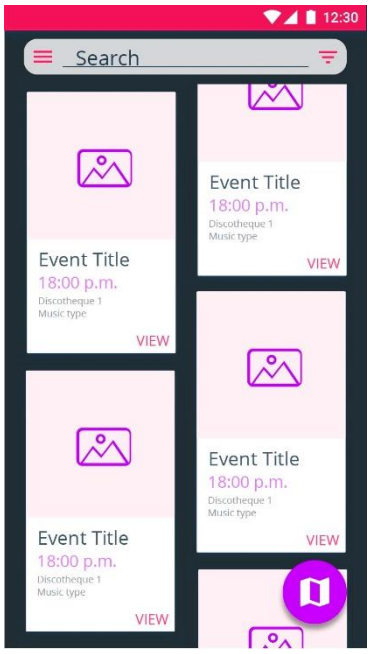
React Native. Facebook [online]. Disponible: https://facebook.github.io/react-native/

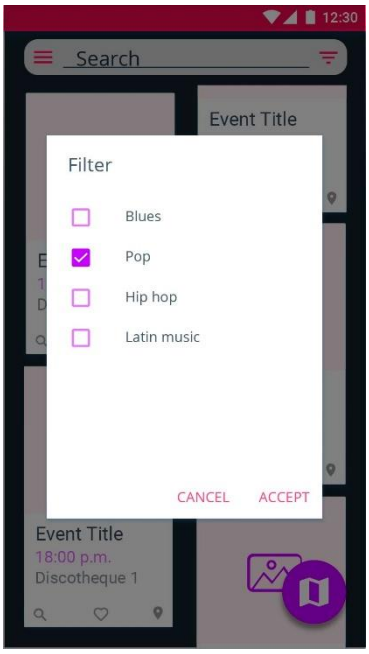
# 6. Anexos

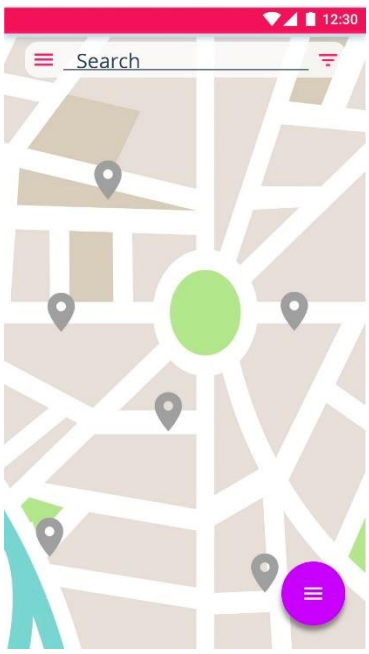
# Anexo I – Pantallas Mockup

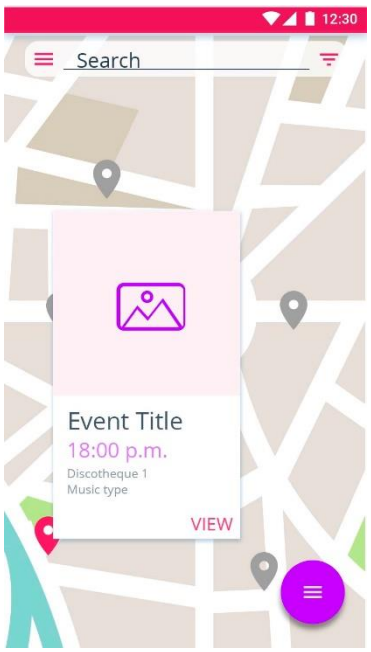



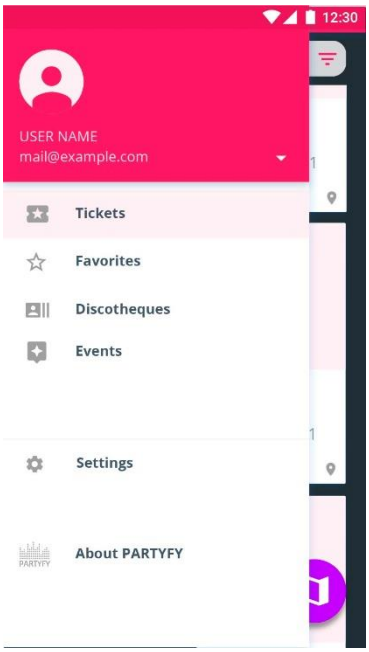



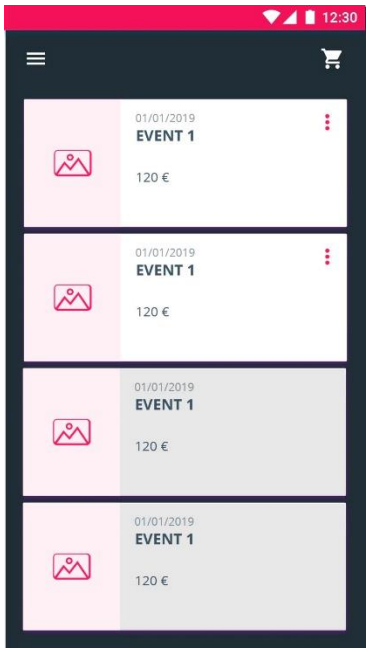



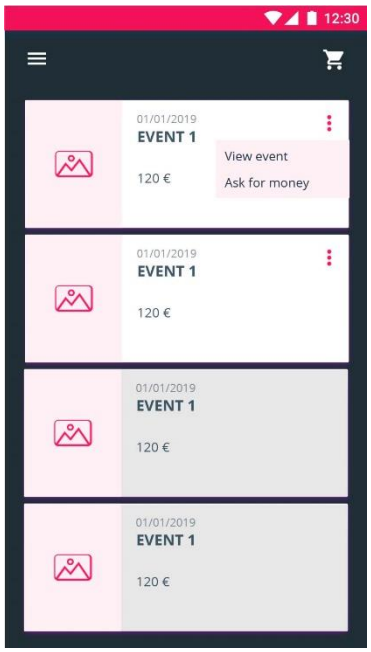



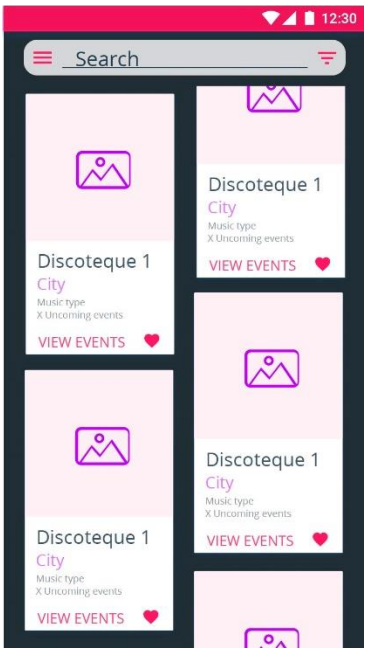

12:30
Search
Event Title
18:00 p.m.
Discotheque 1
Music type
VIEW

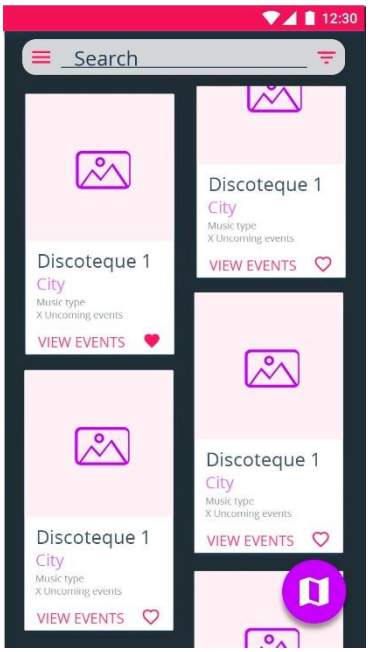

12:30
USER NAME
mail@example.com
Tickets
Favorites
Discotheques
Events
Settings
About PARTYFY

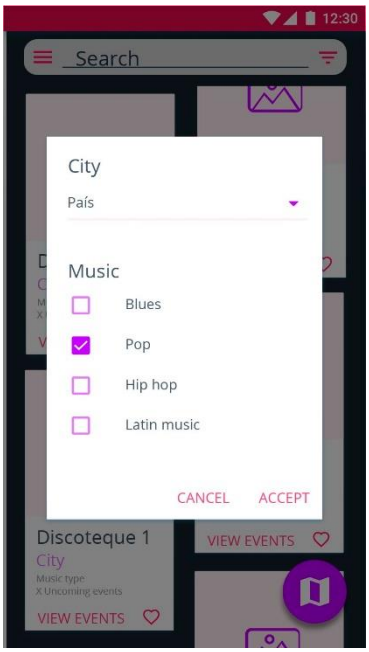

12:30
01/01/2019
EVENT 1
120 €
01/01/2019
EVENT 1
120 €
01/01/2019
EVENT 1
120 €
01/01/2019
EVENT 1
120 €


12:30
01/01/2019
EVENT 1
120 €
View event
Ask for money
01/01/2019
EVENT 1
120 €
01/01/2019
EVENT 1
120 €
01/01/2019
EVENT 1
120 €


12:30

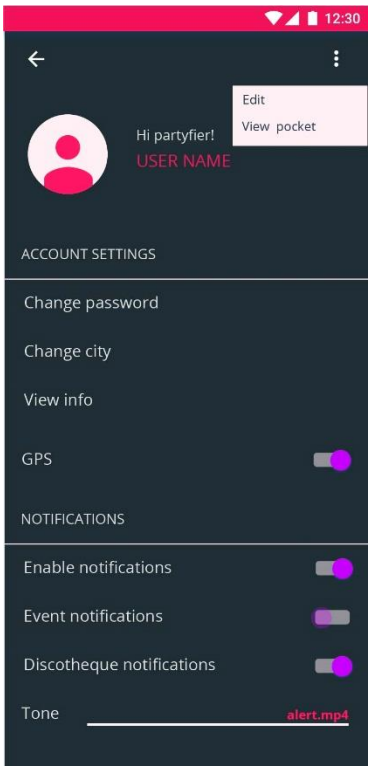

12:30
Search
Discoteque 1
City
Music type
X Uncoming events
VIEW EVENTS
Discoteque 1
City
Music type
X Uncoming events
VIEW EVENTS
Discoteque 1
City
Music type
X Uncoming events
VIEW EVENTS
Discoteque 1
City
Music type
X Uncoming events
VIEW EVENTS

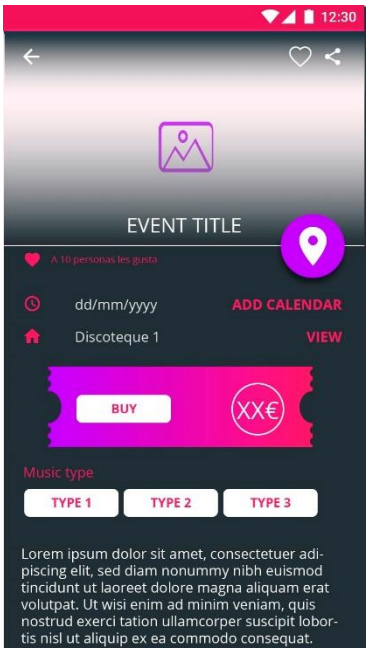

12:30
Search
Discoteque 1
City
Music type
X Uncoming events
VIEW EVENTS

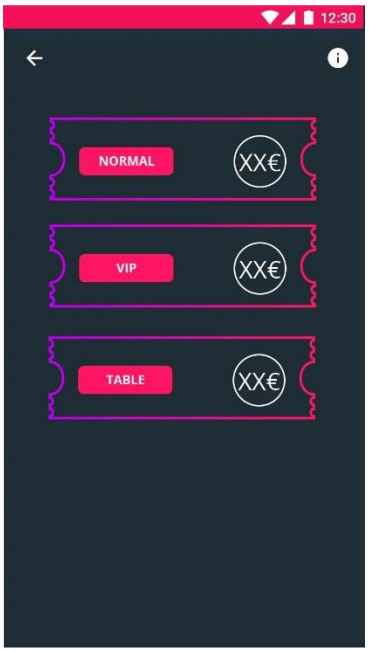

12:30
Search
City
País
Music
Blues
Pop
Hip hop
Latin music
CANCEL
ACCEPT
Discoteque 1
City
VIEW EVENTS

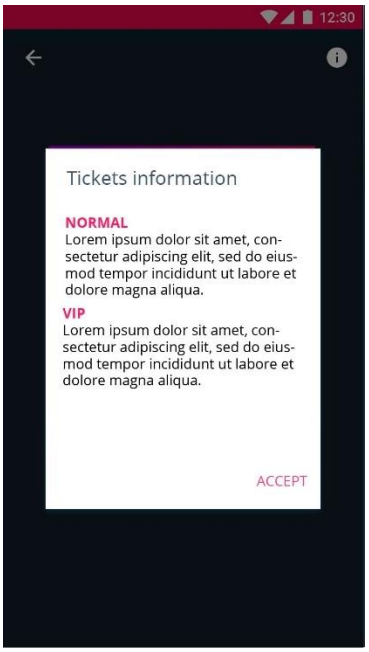

12:30
Search
Discoteque 1
City
Music type
X Uncoming events
VIEW EVENTS

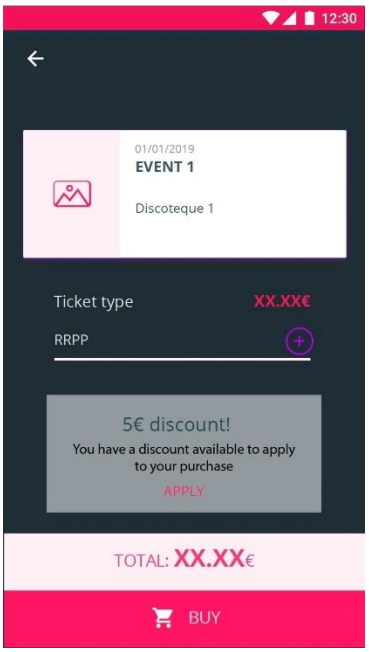

12:30
Jueves 11:00
Viernes 11:00
Sábado 11:00
Music type
Lorem ipsum dolor sit amet, consectetuer adi-piscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobor-tis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel
VIEW EVENTS

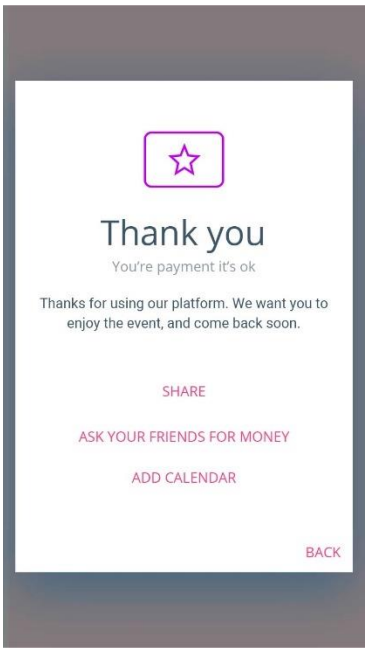

12:30
Edit
View pocket
Hi partyfier!
USER NAME
ACCOUNT SETTINGS
Change password
Change city
View info
GPS
NOTIFICATIONS
Enable notifications
Event notifications
Discotheque notifications
Tone
alert.mp4


12:30
EVENT TITLE
dd/mm/yyyy
ADD CALENDAR
Discoteque 1
VIEW
BUY
XX€
Music type
TYPE 1
TYPE 2
TYPE 3
Lorem ipsum dolor sit amet, consectetuer adi-piscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobor-tis nisl ut aliquip ex ea commodo consequat.


12:30
NORMAL
XX€
VIP
XX€
TABLE
XX€


12:30
Tickets information
NORMAL
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
VIP
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
ACCEPT


12:30
01/01/2019
EVENT 1
Discoteque 1
Ticket type
XX.XX€
RRPP
5€ discount!
You have a discount available to apply to your purchase
APPLY
TOTAL: XX.XX€
BUY


Thank you
You're payment it's ok
Thanks for using our platform. We want you to enjoy the event, and come back soon.
SHARE
ASK YOUR FRIENDS FOR MONEY
ADD CALENDAR
BACK

# Anexo 2 – Índice de ilustraciones